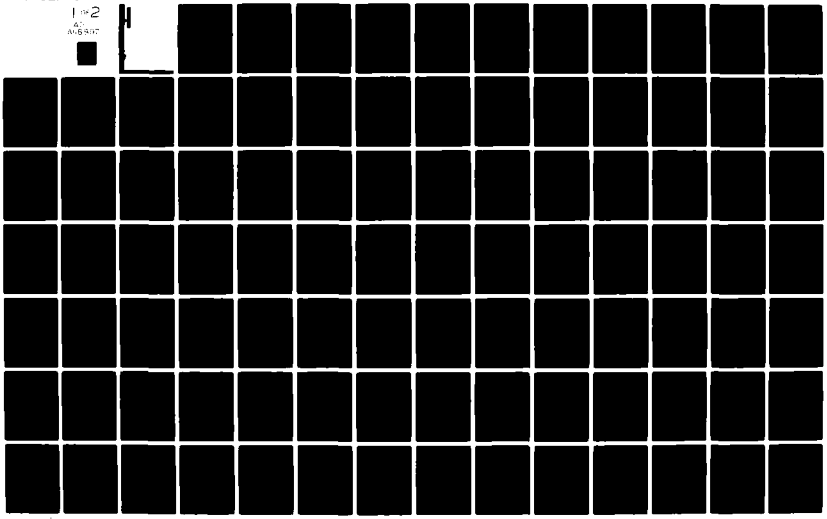
AD-A118 897 NAVAL RESEARCH LAB WASHINGTON DC F/6 9/2
INTERFACE SPECIFICATIONS FOR THE A-7E SHARED SERVICES MODULE.(U)
SEP 82 P C CLEMENTS
UNCLASSIFIED NRL-MR-4863 NL

1 of 2
AD
A118897



AD A118897

Release 1 SS.PNL

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER NRL Memorandum Report 4863	2. GOVT ACCESSION NO. AD-A118 897	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) INTERFACE SPECIFICATIONS FOR THE A-7E SHARED SERVICES MODULE		5. TYPE OF REPORT & PERIOD COVERED Interim report on a continuing NRL problem.
		6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) P.C. Clements		8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS Naval Research Laboratory Washington, DC 20375		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS 62721N; XF21242101; 75-0106-0-2
11. CONTROLLING OFFICE NAME AND ADDRESS		12. REPORT DATE September 8, 1982
		13. NUMBER OF PAGES 106
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)		15. SECURITY CLASS. (of this report) UNCLASSIFIED
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Abstract interfaces	Modules
Avionics software	Real-time systems
Information hiding	Software maintenance
Modular decomposition	Software specifications

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

This document describes the programmer interface to one of the modules in NRL's Software Cost Reduction (SCR) project, which is demonstrating the feasibility of applying advanced software engineering techniques to complex real-time systems in order to simplify maintenance. To illustrate the principles, the onboard software for the Navy's A-7E aircraft will be redesigned and rewritten. The Shared Services

(Continues)

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE
S/N 0102-014-6601

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

20. ABSTRACT (Continued)

module provides those values and services that are used in the computation or derivation of output affecting more than one peripheral device in the system. The purpose of the Shared Services module is to allow the remainder of the software to remain unchanged when the requirements-based rules for these values and services change.

This report describes the modular structure of the Shared Services module, and contains the abstract interface specifications for all of the facilities provided to users. It serves as development and maintenance documentation for the SCR software design, and is also intended as a model for other people interested in applying the abstract interface approach on other software projects.

CONTENTS

INTRODUCTION 1

INTERFACE SPECIFICATIONS

Mode Determination submodule: SS.MODE 3

Panel I/O Support submodule: SS.PNL 6
- Panel Configuration: SS.PNL.CONFIG 7
- Display Format: SS.PNL.FORMAT 10
- Panel Input: SS.PNL.INPUT 14

Shared Subroutine submodule: SS.SUBRTN 21

Stage Director submodule: SS.STAGE 23

System Values submodule: SS.SYSVAL 28
- Device Reasonableness: SS.SYSVAL.DEVREAS 29
- IMS Alignment: SS.SYSVAL.IMSALN 31
- Reference Point: SS.SYSVAL.REFPT 34
- Slew: SS.SYSVAL.SLEW 42
- Value Selection: SS.SYSVAL.VALSEL 45
- Weapon Release: SS.SYSVAL.WPNRLS 48

APPENDIX 1 -- INTERFACE DESIGN ISSUES 53

APPENDIX 2 -- IMPLEMENTATION NOTES 59

APPENDIX 3 -- INTERFACE TERM INDEX WITH MAPPING TO REQUIREMENTS. . 97

ACKNOWLEDGMENTS 103

REFERENCES 103

INTERFACE SPECIFICATIONS FOR THE A-7E SHARED SERVICES MODULE

INTRODUCTION

OVERVIEW

This document describes the interface to the A-7E Shared Services Module. It is one in a series of design documents of the Naval Research Laboratory's Software Cost Reduction (SCR) project, which is producing a fully worked-out example of model software design and documentation based on state-of-the-art software engineering principles.

The role of the Shared Services module in the A-7E software is explained fully in [MG]. Briefly, it is responsible for providing values and services that would otherwise have to be produced by more than one Function Driver module (described in [FD]). Its primary purpose is to prevent a duplication of effort in design, implementation, and execution.

The design of this module was carried out in concert with the specification of the Function Driver module. Because the Function Driver specifications consist mainly of tables of conditions and events (as explained in [REQ], Section 0.1), it was usually a straightforward matter to list those calculations which would be necessary for each function driver submodule to produce a device-driving value. From that list, commonalities could be extracted, and facilities to provide the common values/services could be easily constructed.

READER PREREQUISITIES

It is assumed that the reader is somewhat familiar with [FD], and more familiar with [SO], [MG] and [REQ].

Manuscript submitted May 13, 1982.

STRUCTURE OF THE MODULE

The module has five submodules:

Mode Determination submodule: This submodule is responsible for keeping track of and reporting the current mode(s) of the system. Some modes are builtin; these are described Chapter 3 of [REQ]. Facilities are also provided for users to define new modes based on combinations of previously defined modes.

Panel I/O Support submodule: This submodule provides service routines that facilitate simplified input and output operations on the A-7 computer panel.

Shared Subroutine submodule: This submodule provides some mathematical routines that are required by more than one function driver module.

Stage Director submodule: Some system modes are divided into stages that are sequenced through; a stage transition occurs when some chosen goal is reached. This submodule tracks and reports the current stage(s) of the system. Stages are discussed in Chapter 3 of [REQ].

System Values submodule: This submodule provides values that are used by more than one function driver module, or values whose rules for computation are similar enough that it is profitable to compute them together.

STANDARD ORGANIZATION

The standard organization of each interface description is described in [SO], subject to the following variations.

The "Implementation Notes" for each interface description often contain event tables or condition tables or selector tables. The semantics of these tables are explained in Section 0.3 of [REQ].

The "Dictionary" and/or "Implementation Notes" section of each interface description may contain a term or terms enclosed in "!!" brackets. This denotes a term whose value is not reported out by the module; rather, it is a term that may be used to replace a lengthy (but precise) definition used more than once in that particular interface description.

Mode Determination Submodule

SS.MODE.1 Introduction

The run-time state of the A-7 flight program can be partially characterized by its current operating modes. This module provides facilities for accessing current modes in two ways: (1) it provides an access program that, given a mode name, returns true if the system is currently in the specified mode, and (2) it signals changes in modes. In addition, the module provides facilities for defining new modes in terms of previously defined modes.

SS.MODE.2 Interface Overview

SS.MODE.2.1 DECLARATION OF MODE NAMES

Program name	Parm type	Parm info	Undesired events
++DCL_MODE++	p1:modename;I	new mode name	%undefined mode%
	p2:modelist;I	defining modes	%mode already defined%

Program effects

Declares the identifier given in p1 to be an alias for each of the previously defined modes listed in p2. Modes that have been declared may be used as operands in subsequent code. In addition, the event of mode entry and the event of mode exit will be signaled for declared modes. Where a mode is defined in terms of a set of modes, the system enters the mode when the system enters any mode in the set, and the system is not already in some mode in the set; the system exits the mode when the system exits a mode in the set and is in no other mode in the set.

SS.MODE.2.2 OPERATIONS ON MODES

Program name	Parm type	Parm info	Undesired events
+IN_MODE+	p1:modename;I	mode name	%undefined mode%
	p2:boolean;O	in mode?	

Program effects

p2 := true iff the system is currently in mode p1.

SS.MODE.2.3 Events signalled

@T(!+in _____+!)   @F(!+in _____+!)

where "_____" is the name of any mode.

SS.MODE.3.1 Basic Assumptions

1. The system is always in at least one mode. Changes in current modes can always be signalled to user programs.

2. This module can always determine if the system is in a particular mode.

3. Mode transitions may be considered to be instantaneous.

SS.MODE.3.2 Undesired Event Assumptions

1. User programs will not use an undefined mode as an operand.

3. User programs will not define the same mode name more than once.

SS.MODE.4 Local Type Definitions

modename	An identifier to be used as the name of a mode.
modelist	a sequence of def_modes enclosed in parentheses and separated by commas. For instance: (*Sautocal*, *SINSaln*, *DIG*, *Airaln*, *DI*)
def_mode	a mode name defined in a previous ++DCL_MODE++ operation, or one of the pre-defined modes given in [REQ], Sections 3.0.1 and 3.6.16.

SS.MODE.5 Local Dictionary

!+in _____+!	(where "_____" must be replaced by a mode name) True if the system is in the named mode and false otherwise.

SS.MODE.6 Undesired Event Dictionary

%undefined mode%	A user program has specified an alias for a non-existent mode.
%mode already defined%	A user program has attempted to redefine a mode name.

SS.MODE.7 System Generation Parameters: None.

SS.MODE.8 Information Hidden

1. Representation of the correspondence between defined mode names and the defining modes.
2. Algorithm for signalling mode changes.
3. What causes transitions among the Requirements-defined modes. How the mode transition criteria is represented.

Panel I/O Support Submodule

This submodule is responsible for providing services for programs using the computer panel. Such services facilitate simpler input and output operations than would be possible using the panel's abstract interface.

The submodule is further decomposed into the following submodules:

Configuration submodule: responsible for detecting and reporting the current control configuration of the panel. The control configuration determines what should be currently displayed on the panel's upper and lower data windows, as well as what value will be updated should an input operation occur.

Display format submodule: provides requirements-oriented display formats (such as angle and latitude) so that user programs need not be concerned with how such quantities are actually represented on the panel.

Input submodule: responsible for accepting and reporting all input values entered by the pilot through the panel. Hides the input protocols, and the rules for determining what value is being updated by the current input operation.

Panel I/O Support:

Panel Configuration Submodule

SS.PNL.CONFIG.1 Introduction

This submodule is responsible for determining and reporting the current control configuration of the panel. The configuration is determined by the state of various switches, controls, and conditions. The control configuration may be used to determine what should be displayed on the panel, or what value will be updated should an input operation occur.

SS.PNL.CONFIG.2. Interface Overview

SS.PNL.CONFIG.2.1 Access Function Table

Access program name	Parm type	Parm info	UE:
+G_CONTROL_CONFIG+	p1:panel_config;I		None.
	p2:boolean;O	!+in pnl config+!	

SS.PNL.CONFIG.2.2 Events Signalled

@T(!+pnl config changed+!)

@T(!+pnl config+! = xxx) where 'xxx' may be replaced by any member of the domain of type panel_config, as enumerated under section SS.PNL.CONFIG.5.

SS.PNL.CONFIG.3 Basic Assumptions:

1. There are controls that affect what should be displayed on the panel at a given time. The state of these controls is known as the panel control configuration, and is detectable to this module.

2. The panel can be in at most two control configurations simultaneously.

3. User programs can consider transitions between configurations to be instantaneous.

4. The panel control configuration also affects how data entered through the panel is interpreted.

SS.PNL.CONFIG.4 Local Data Types:

Type	Type definition
panel_config	Enumerated by the following list:

$align_stage$	$alt AGL at rls$
$alt baro AGL$	$ARPINT$
$ARPQUANT$	$az miss dist at rls$
$az ref hdg$	$burst ht$
$central long a$	$central long b$
$compfail$	$data nbr$
$dest altitude$	$dest lat$
$dest long$	$dest mslp$
$Doppler coupled$	$drift angle filtered$
$drftangl IMS$	$e coarse bias$
$e coarse scale$	$e fine bias$
$e fine scale$	$elapsed navaln time$
$fpangl at rls$	$gndspd filtered$
$groundspeed IMS$	$gyro drift delta n$
$heading IMS$	$hdg system$
$heading MAG$	$IMS diags1$
$IMS diags2$	$IMS total vel$
$L-probe$	$land based$
$latitude$	$latitude error$
$longitude$	$longitude error$
$low lat ct a$	$low lat ct b$
$mag variation$	$map latitude$
$map longitude$	$map orient a$
$map orient b$	$map sw diags$
$mark lat$	$mark long$
$MFSW diags$	$n coarse bias$
$n coarse scale$	$n fine bias$
$n fine scale$	$nav diags1$
$nav diags2$	$norm accel at rls$
$offset brg$	$offset dht$
$offset rng$	$OFP ver1$
$OFP ver2$	$priority alt display$
$radalt priority$	$SINS dhdg$
$SINS east vel$	$SINS heading$
$SINS lat$	$SINS long$
$SINS north vel$	$SINS valid1$
$SINS valid2$	$SINS x offset$
$SINS y offset$	$SINS z offset$
$slant range at rls$	$STARDY diags$
$TAS ADC at rls$	$TAS filtered$
$time to dest$	$v coarse bias$
$v coarse scale$	$vel e$
$vel n$	$WEAPTYP$
$wind dir$	$wind vel$
$wpn sw diags$	$x corr increm$
$x drift$	$y corr increm$
$y drift$	$z corr increm$
$z drift$	$none$

SS.PNL.CONFIG.5 Dictionary

!+pnl config changed+!	true while !+pnl config+! is changing value.
!+in pnl config+!	true iff the panel is in the control configuration given by p1.
!+pnl config+!	The current panel control configuration.

SS.PNL.CONFIG.6 Undesired Event Dictionary: None

SS.PNL.CONFIG.7 System Generation Parameters: None

SS.PNL.CONFIG.8 Information Hidden

1. The data number associated with most of the display items.

2. Which configurations occur simultaneously.

3. The way the various panel switches affect the panel control configurations; which configurations require keyboard input, and what input is required in such cases.

Panel I/O Support:

Panel Display Format Submodule

SS.PNL.FORMAT.1 Introduction

This submodule is responsible for providing formatting services for all data displayed on the panel. Its secret is how the various formats are actually displayed, using the virtual panel interface provided by the Device Interface Module.

SS.PNL.FORMAT.2. Interface Overview

SS.PNL.FORMAT.2.1 Access Program Table

In the table, "win" may be replaced by "UPPER" or "LOWER".

Program name	Parm type	Parm info	UEs
+S_ANGLE_win+	p1:angle;I		None.
+S_BITSTRING_win+	p1:bitstring;I		
+S_BINT_win+	p1:integer;I		
+S_BLNKLTS_win+	--		
+S_CHARSTR_LOWER+	p1:charstr;I		
+S_CHARSTR_UPPER+	p1:charstr;I		
+S_LATITUDE_win+	p1:latitude;I		
+S_LONGITUDE_win+	p1:longitude;I		
+S_REAL_win+	p1:real;I		
+S_SIGN_win+	p1:boolean;I		
+S_SFRAC_win+	p1:real;I		
+S_SINT_win+	p1:integer;I		
+S_S2INT_win+	p1:integer;I		
+S_TIME_win+	p1:time;I		
+S_UFRAC_win+	p1:real;I		
+S_UINT_win+	p1:integer;I		
+CLEAR_win+	--		

Program Effects

All programs erase the previous display in the affected window. In addition:

+CLEAR_LOWER+ +CLEAR_UPPER+	Turns off all the format lights associated with that window.
+S_BLNKLTS_LOWER+ +S_BLNKLTS_UPPER+	Turns on all format lights associated with that window.
+S_BITSTRING_win+ +S_BINT_win+ +S_CHARSTR_win+ +S_REAL_win+	Displays pl in the specified window. If the value of pl exceeds the display capacity of the window, then the value is truncated on the right (least significant) side. If the value of pl takes less than the length of the window, than the value will be displayed right justified, with blank fill on the left.
+S_SIGN_win+	Displays pl in the specified window.
+S_ANGLE_win+ +S_LATITUDE_win+ +S_LONGITUDE_win+	Displays pl in the specified window in degrees/minutes/seconds form.
+S_SFRAC_win+	Displays a signed real number whose absolute value is less than one. If a number of larger magnitude is given, only the fractional part will be displayed. Number is left justified, with zero fill on right.
+S_SINT_win+	Displays pl, with sign, in the given window, with zero fill to the left.
+S_S2INT_win+	Displays a signed two-digit integer in the given window, with zero fill to the right. If pl has more than two digits, it will be truncated on the right.
+S_TIME_win+	Displays a time in the given window, in hours/minutes/seconds form.

Program effects (continued)

+S_UFRAC_win+	Displays an unsigned real number whose absolute value is less than one. If a number of larger magnitude is given, only the fractional part will be displayed. If a negative number is given, the absolute value will be taken. Number is left justified, with zero fill on right.
+S_UINT_win+	Displays the magnitude of p1 in the given window, without sign, and with zero fill to left.

SS.PNL.FORMAT.2.2 Events Signalled: None.

SS.PNL.FORMAT.3 Basic Assumptions:

1. Data must be displayed on the panel using one of the following display formats: ANGLE, BITSTRING, INTEGER, CHARACTER STRING, LATITUDE, LONGITUDE, REAL, SIGN, SIGNED FRACTION, SIGNED INTEGER, UNSIGNED FRACTION, UNSIGNED INTEGER, SIGNED TWO-DIGIT INTEGER, or TIME. There is only one legal data type for each panel format. A format is defined by which elements of a window are used, and which associated format lights are turned on and off.

2. When data is displayed in a window, any previous display in that window is erased.

3. If the data to be displayed in a window exceeds the display capacity of that window, then the data will be truncated on the right (least significant) side.

4. There is a format known as BLANK LIGHTS that may be displayed on the panel. No data is displayed when this format is invoked. All format lights associated with a window are turned on, and the window is blanked.

5. If control of the panel is not available to this module when a display command is issued, then the command will be held until control of the panel is available and displayed then. If more than one display command is received when the panel is not available, then the most recent one will be carried out when the panel becomes available.

SS.PNL.FORMAT.4 Local Data Types: None.

SS.PNL.FORMAT.5 Dictionary: None.

SS.PNL.FORMAT.6 Undesired Event Dictionary: None.

SS.PNL.FORMAT.7 System Generation Parameters: None.

SS.PNL.FORMAT.8 Information Hidden:

1. How the interface of the DIM virtual panel is used to implement the services provided by this submodule.

2. How the format lights are used to display certain data formats.

3. Certain format rules that govern how values are displayed, such as: how signs (positive/negative) are displayed; how bitstrings and booleans are displayed; when decimal points are used, and when they are merely implied.

Panel I/O Support:

Panel Input Submodule

SS.PNL.INPUT.1 Introduction

This submodule is responsible for accepting all input keyed in by the pilot through the panel. It contains the knowledge of what constitutes a successful or unsuccessful input operation, and the procedures and protocol for entering data. This module also knows the list of items whose values can be updated by pilot entry, and can tell which item is being updated. Finally, it can at any time provide the most recently-entered value for any such item.

SS.PNL.INPUT.2. Interface Overview

SS.PNL.INPUT.2.1 Access Program Table

Program name	Parm type	Parm info	Undesired Events
		Restricted device reconfiguration value programs	
+G_CENTRAL_LONG_A+	p1:longitude;O	!+central long a pnl+!	None.
+G_CENTRAL_LONG_B+	p1:longitude;O	!+central long b pnl+!	
+G_E_COARSE_BIAS+	p1:real;O	!+e coarse bias pnl+!	
+G_E_COARSE_SCALE+	p1:real;O	!+e coarse scale pnl+!	
+G_E_FINE_BIAS+	p1:real;O	!+e fine bias pnl+!	
+G_E_FINE_SCALE+	p1:real;O	!+e fine scale pnl+!	
+G_L_PROBE+	p1:boolean;O	!+L-probe pnl+!	
+G_LOW_LAT_CT_A+	p1:integer;O	!+low lat ct a pnl+!	
+G_LOW_LAT_CT_B+	p1:integer;O	!+low lat ct b pnl+!	
+G_MAP_ORIENT_A+	p1:angle;O	!+map orient a pnl+!	
+G_MAP_ORIENT_B+	p1:angle;O	!+map orient b pnl+!	
+G_N_COARSE_BIAS+	p1:real;O	!+n coarse bias pnl+!	
+G_N_COARSE_SCALE+	p1:real;O	!+n coarse scale pnl+!	
+G_N_FINE_BIAS+	p1:real;O	!+n fine bias pnl+!	
+G_N_FINE_SCALE+	p1:real;O	!+n fine scale pnl+!	
+G_V_COARSE_BIAS+	p1:real;O	!+v coarse bias pnl+!	
+G_V_COARSE_SCALE+	p1:real;O	!+v coarse scale pnl+!	
+G_X_CORR_INCREM+	p1:real;O	!+x corr increm pnl+!	
+G_X_DRIFT+	p1:real;O	!+x drift pnl+!	
+G_Y_CORR_INCREM+	p1:real;O	!+y corr increm pnl+!	
+G_Y_DRIFT+	p1:real;O	!+y drift pnl+!	
+G_Z_CORR_INCREM+	p1:real;O	!+z corr increm pnl+!	
+G_Z_DRIFT+	p1:real;O	!+z drift pnl+!	

SS.PNL.INPUT.2.1 Access Program Table (continued):

Program name	Parm type	Parm info	Undesired Events
		Parameterized items	
+G_BURST_HT+	p1:integer;I	locn nbr	%locn nbr illegal%
	p2:distance;O	!+burst ht pnl+!	
+G_DEST_ALTITUDE+	p1:integer;I	locn nbr	
	p2:distance;O	!+dest altitude pnl+!	
+G_DEST_LAT+	p1:integer;I	locn nbr	
	p2:latitude;O	!+dest lat pnl+!	
+G_DEST_LONG+	p1:integer;I	locn nbr	
	p2:longitude;O	!+dest long pnl+!	
+G_DEST_MSLP+	p1:integer;I	locn nbr	
	p2:pressure;O	!+dest mslp pnl+!	
+G_MAG_VARIATION+	p1:integer;I	locn nbr	
	p2:angle;O	!+mag variation pnl+!	
+G_OFFSET_BRG+	p1:integer;I	locn nbr	
	p2:angle;O	!+offset brg pnl+!	
+G_OFFSET_DHT+	p1:integer;I	locn nbr	
	p2:distance;O	!+offset dht pnl+!	
+G_OFFSET_RNG+	p1:integer;I	locn nbr	
	p2:distance;O	!+offset rng pnl+!	
		Other Panel Input Items	
+G_AZ_REF_HDG+	p1:angle;O	!+az ref hdg pnl+!	None.
+G_DATA_NBR+	p1:integer;O	!+data nbr pnl+!	
+G_DEST_ENTRY+	p1:integer;O	!+dest entry pnl+!	
+G_DOPPLER_COUPLED+	p1:boolean;O	!+Doppler coupled pnl+!	
+G_LAND_BASED+	p1:boolean;O	!+land based pnl+!	
+G_LATITUDE+	p1:latitude;O	!+latitude pnl+!	
+G_LONGITUDE+	p1:longitude;O	!+longitude pnl+!	
+G_RADALT_PRIORITY+	p1:boolean;O	!+radalt priority pnl+!	
+G_SINS_DHDG+	p1:angle;O	!+SINS dhdg pnl+!	
+G_SINS_X_OFFSET+	p1:distance;O	!+SINS x offset pnl+!	
+G_SINS_Y_OFFSET+	p1:distance;O	!+SINS y offset pnl+!	
+G_SINS_Z_OFFSET+	p1:distance;O	!+SINS z offset pnl+!	
+G_WIND_DIR+	p1:angle;O	!+wind dir pnl+!	
+G_WIND_VEL+	p1:speed;O	!+wind vel pnl+!	

SS.PNL.INPUT.2.2 Events Signalled

@T(!+data enterable+!)	@F(!+data enterable+!)
@T(!+dest selected+!)	
@T(!+input attempted+!)	
@T(!+input requested+!)	
@T(!+land based+!)	@F(!+land based+!)
@T(!+panel error+!)	
@T(!+pnl input complete+!)	

@T(!+new "data" entered+!)
where "data" may be replaced by any interface term in the Parm Info column of the access program table overview, minus its brackets. For instance, @T(!+new central long a pnl entered+!) is signalled, as are @T(!+new burst ht pnl entered+!) and @T(!+new wind dir pnl entered+!).

SS.PNL.INPUT.3.1 Basic Assumptions:

1. There are well-defined protocols and procedures for entering data through the panel. These protocols determine when data may be entered, order of operations and size/scale of entered values. For some input operations, it is necessary for the pilot to issue a preliminary keyboard command to begin the operation.

2. When input protocols are violated, an error display is shown for a short fixed period of time, and the input operation is considered unsuccessful. Nothing else may be displayed on the panel while the error display is being shown. When the error display is removed, the panel displays what would have been displayed had there been no attempted input operation.

3. This submodule controls the virtual panel during an input operation.

4. Only one input operation may be in progress at a time. It is possible to determine when an input operation has begun and when it has ended. This module reports when a data input operation may legally begin, when one has begun, when one has successfully terminated, and when a violation of input rules has occurred.

5. It is possible to interpret panel inputs as assignments of value to particular items. For a particular item, the value returned by this module is the most-recently-entered value of that item. If any item has not been assigned a value by panel input since the program was loaded, a default value will be returned. The default values are given as system generation parameters.

6. Some input items have more than one value associated with them. For instance, destination latitude is such an item; it may have several values (one for each of several destinations) associated with it. The set of such multiple-value panel input items is fixed and determined by requirements. When a pilot enters a value for such an item, he also provides an integer that specifies which particular value for that item is being entered. To retrieve a particular value of a multiple-value item, user programs must specify the same integer as that provided by the pilot when the value was entered. A string of consecutive integers constitutes the only legal integer values. The upper and lower bound of this string are fixed at system generation time.

7. Any program in this submodule may be called at any time. An input operation may occur at any time, and is not under the control of any user program.

SS.PNL.INPUT.3.2 Assumptions about Undesired Events:

1. User programs will not call attempt to retrieve a multiple-value input item by specifying an illegal integer identifier.

SS.PNL.INPUT.4 Local Data Types: None

SS.PNL.INPUT.5 Dictionary

Term	Definition
GENERAL DEFINITIONS:	
Any item of the form !+"data" pnl+!:	The last value of !+"data"+! entered via the panel, where each such item is defined in the Data Banker dictionary (unless defined below). For instance, !+latitude pnl+! is the last value of !+latitude+! entered via the panel. If, for a particular item, no value has been entered via the panel since the program was loaded, a default value will be returned. The default value for each item is given by a system generation parameter.
any item of the form !+new "data" entered+!	signalled when the pilot has just entered a new value for !+"data" pnl+!, which is defined for each "data" elsewhere in this dictionary.
OTHER DEFINITIONS:	
!+data enterable+!	true iff a panel input operation may legally begin.
!+dest selected+!	true while !+dest entry pnl+! is changing value.
!+input attempted+!	true while the pilot is attempting to enter input through the panel without issuing a preliminary keyboard command to begin the operation.
!+input requested+!	true while the pilot is issuing the preliminary keyboard command to begin an input operation.
!+land based+!	The most recently-entered value for !+land based pnl+!.
!+panel error+!	true while the panel is displaying the error message.
!+pnl input complete+!	true between the time that one panel input operation has terminated normally and the time that another panel input operation has begun.

SS.PNL.INPUT.6 Undesired Event Dictionary:

%locn nbr illegal%	pl lt #multval_lbound# OR pl gt #multval_hbound#.

SS.PNL.INPUT.7 System Generation Parameters:

The following system generation parameters give the load-time initial values of the items indicated:

System generation parameter	Initial value of
#pnl init burst ht#	!+burst ht pnl+!
#pnl init dest altitute#	!+dest altitude pnl+!
#pnl init dest lat#	!+dest lat pnl+!
#pnl init dest long#	!+dest long pnl+!
#pnl init dest mslp#	!+dest mslp pnl+!
#pnl init mag variation#	!+mag variation pnl+!
#pnl init offset brg#	!+offset brg pnl+!
#pnl init offset dht#	!+offset dht pnl+!
#pnl init offset rng#	!+offset rng pnl+!
#pnl init az ref hdg#	!+az ref hdg pnl+!
#pnl init data nbr#	!+data nbr pnl+!
#pnl init dest entry#	!+dest entry pnl+!
#pnl init Doppler coupled#	!+Doppler coupled pnl+!
#pnl init land based#	!+land based pnl+!
#pnl init latitude#	!+latitude pnl+!
#pnl init longitude#	!+longitude pnl+!
#pnl init radalt priority#	!+radalt priority pnl+!
#pnl init SINS dhdg#	!+SINS dhdg pnl+!
#pnl init SINS offset#	!+SINS x offset pnl+!, !+SINS y offset pnl+!, and !+SINS z offset pnl+!
#pnl init wind dir#	!+wind dir pnl+!
#pnl init wind vel#	!+wind vel pnl+!

#error display time#	How long to display the error message. At the end of this time, the error display is removed.
#error msg upper#	A character string of length 6; this is used as that part of the error message displayed in the upper window.
#error msg lower#	A character string of length 7; this is used as that part of the error message displayed in the lower window.
#multval_hbound#	The highest legal integer associated with a multiple-value panel input item.
#multval_lbound#	The lowest legal integer associated with a multiple-value panel input item.

SS.PNL.INPUT.8 Information Hidden:

1. The sequence of operations necessary to perform panel input; how to tell when a panel input operation has started and terminated.

2. When and how each panel input operation is performed; how long ago each input operation occurred.

3. How this module makes use of other panel i/o submodules to tell what value is being updated by any input operation.

4. How the pilot enters the identifying integer for multiple-value panel input items; whether he employs the panel (and if so, whether or not the integer value is provided as a separate input item), or some other means.

Shared Subroutine Submodule

SS.SUBRTN.1 Introduction

This module provides a few mathematical service routines that are required by more than one Function Driver module. Each value provided is a pre-defined arithmetic function of the input parameters.

SS.SUBRTN.2. Interface Overview

SS.SUBRTN.2.1 Access Program Table

Program name	Parm type	Parm info	UEs
+LIMIT_FN+	p1:angle;I		None.
	p2:real;I		
	p3:angle;I		
	p4:angle;O	!+limited input+!	
+SYMBOL_AZ_ON_ASL+	p1:angle;I	symbol elevation	
	p2:angle;O	!+symbol_az_on_ASL+!	

SS.SUBRTN.2.2 Events Signalled: None.

SS.SUBRTN.3 Basic Assumptions:

1. Some mathematical calculations must be performed by more than one Function Driver module.

2. It is possible to determine the azimuth angle of a point on the HUD azimuth steering line (ASL) by giving the elevation of the point, provided the position and rotation angle of the ASL are also available.

SS.SUBRTN.4 Local Data Types: None.

SS.SUBRTN.5 Dictionary

!+limited input+!	SIGN(p1) x MIN(p3, p2 x ABS(p1))
!+symbol_az_on_ASL+!	The azimuth angle of a point on the HUD azimuth steering line (ASL) at a given elevation.

SS.SUBRTN.6 Undesired Event Dictionary: None.

SS.SUBRTN.7 System Generation Parameters: None.

SS.SUBRTN.8 Information Hidden

1. The algorithms used to compute the output values.

Stage Director Submodule

SS.STAGE.1 Introduction

In some system modes, the system sequences through stages where the end of each stage is marked by the achievement of some chosen goal and the next stage is directed towards achieving a new goal. The modes differ in the definition of the goals and the sequence of stages. The Stage Director module provides information about the current stage of such modes. The secrets of these modules are the rules for choosing the current stage. These rules are properties of the individual modes rather than the individual function drivers. Stage directors do not hide the definitions of conditions that are global to the modes. There are stage directors provided for alignment mode and test mode stages. Alignment modes and test modes are defined in [REQ], Section 3.0.1.

SS.STAGE.2. Interface Overview

SS.STAGE.2.1 Access Program Table

Program name	Parm type	Parm info	UEs
	Alignment Mode Stage Director Programs		
+G_ALIGN_STAGE+	p1:astage;O	!+align_stage+!	None.
+G_CA_STAGE_COMPLETE+	p1:boolean;O	!+CA stage complete+!	
+G_CA2_STAGE_COMPLETE+	p1:boolean;O	!+CA2 stage complete+!	
+G_CL_STAGE_COMPLETE+	p1:boolean;O	!+CL stage complete+!	
+G_CL2_STAGE_COMPLETE+	p1:boolean;O	!+CL2 stage complete+!	
+G_ED_STAGE_COMPLETE+	p1:boolean;O	!+ED stage complete+!	
+G_ED2_STAGE_COMPLETE+	p1:boolean;O	!+ED2 stage complete+!	
+G_FM_STAGE_COMPLETE+	p1:boolean;O	!+FM stage complete+!	
+G_FG_STAGE_COMPLETE+	p1:boolean;O	!+FG stage complete+!	
+G_HG_STAGE_COMPLETE+	p1:boolean;O	!+HG stage complete+!	
+G_HL_STAGE_COMPLETE+	p1:boolean;O	!+HL stage complete+!	
+G_HS_STAGE_COMPLETE+	p1:boolean;O	!+HS stage complete+!	
+G_ND_STAGE_COMPLETE+	p1:boolean;O	!+ND stage complete+!	
+G_ND2_STAGE_COMPLETE+	p1:boolean;O	!+ND2 stage complete+!	
+G_TS_STAGE_COMPLETE+	p1:boolean;O	!+TS stage complete+!	

Program name	Parm type	Parm info	UEs
Test Mode Stage Director Programs			
+G_TEST_STAGE+	pl:tstage;0	!+test_stage+!	None.
+G_AC_STAGE_COMPLETE+	pl:boolean;0	!+AC stage complete+!	
+G_CS_STAGE_COMPLETE+	pl:boolean;0	!+CS stage complete+!	
+G_DC_STAGE_COMPLETE+	pl:boolean;0	!+DC stage complete+!	
+G_DIO_STAGE_COMPLETE+	pl:boolean;0	!+DIO stage complete+!	
+G_GA_STAGE_COMPLETE+	pl:boolean;0	!+GA stage complete+!	
+G_PD_STAGE_COMPLETE+	pl:boolean;0	!+PD stage complete+!	
+G_SC_STAGE_COMPLETE+	pl:boolean;0	!+SC stage complete+!	
+G_TM_STAGE_COMPLETE+	pl:boolean;0	!+TM stage complete+!	

SS.STAGE.2.2 Events Signalled

```
@T(!+align_stage+! = $CA$)        @F(!+align_stage+! = $CA$)
@T(!+align_stage+! = $CA2$)       @F(!+align_stage+! = $CA2$)
@T(!+align_stage+! = $CL$)        @F(!+align_stage+! = $CL$)
@T(!+align_stage+! = $CL2$)       @F(!+align_stage+! = $CL2$)
@T(!+align_stage+! = $ED$)        @F(!+align_stage+! = $ED$)
@T(!+align_stage+! = $ED2$)       @F(!+align_stage+! = $ED2$)
@T(!+align_stage+! = $FM$)        @F(!+align_stage+! = $FM$)
@T(!+align_stage+! = $FG$)        @F(!+align_stage+! = $FG$)
@T(!+align_stage+! = $HG$)        @F(!+align_stage+! = $HG$)
@T(!+align_stage+! = $HL$)        @F(!+align_stage+! = $HL$)
@T(!+align_stage+! = $HS$)        @F(!+align_stage+! = $HS$)
@T(!+align_stage+! = $ND$)        @F(!+align_stage+! = $ND$)
@T(!+align_stage+! = $ND2$)       @F(!+align_stage+! = $ND2$)
@T(!+align_stage+! = $TS$)        @F(!+align_stage+! = $TS$)
@T(!+align_stage+! = $None$)
@T(!+new align stage+!)

@T(!+test_stage+! = $CS$)         @F(!+test_stage+! = $CS$)
@T(!+test_stage+! = $TM$)         @F(!+test_stage+! = $TM$)
@T(!+test_stage+! = $GA$)         @F(!+test_stage+! = $GA$)
@T(!+test_stage+! = $DIO$)        @F(!+test_stage+! = $DIO$)
@T(!+test_stage+! = $SC$)         @F(!+test_stage+! = $SC$)
@T(!+test_stage+! = $DC$)         @F(!+test_stage+! = $DC$)
@T(!+test_stage+! = $AC$)         @F(!+test_stage+! = $AC$)
@T(!+test_stage+! = $PD$)         @F(!+test_stage+! = $PD$)
@T(!+test_stage+! = $None$)
@T(!+new test stage+!)
```

SS.STAGE.3 Basic Assumptions:

1. All alignment modes have stages. An alignment stage is defined by a goal or action, at the completion of which the stage is considered to have terminated. If the system is not in an alignment mode, then the alignment stage will be \$None\$.

2. It is always possible to determine the alignment stage of the system. When the system is in an alignment mode, it will be in exactly one alignment stage.

3. All test modes have stages. A test stage is defined by a goal or action, at the completion of which the stage is considered to have terminated. If the system is not in a test mode, then the test stage will be \$None\$.

4. It is always possible to determine the test stage of the system. When the system is in a test mode, it will be in exactly one test stage.

5. User programs may consider transitions between stages to be instantaneous.

6. Stages may be repeated in the duration of a mode. Not all stages occur in every mode. A particular mode may comprise different stages than it did the last time that mode was entered. The order of stages in a particular mode be different than it was the last time that mode was entered. The stages and their order in a particular invocation of a mode is determined by information available to this module.

SS.STAGE.4 Local Data Types

astage	Enumerated: \$CA\$, \$CA2\$, \$CL\$, \$CL2\$, \$ED\$, \$ED2\$, \$FM\$, \$FG\$, \$HG\$, \$HL\$, \$HS\$, \$ND\$, \$ND2\$, \$TS\$, \$None\$.
tstage	Enumerated: \$CS\$, \$TM\$, \$GA\$, \$DIO\$, \$SC\$, \$DC\$, \$AC\$, \$PD\$, \$None\$.

SS.STAGE.5 Dictionary

!+align_stage+!	The current alignment mode stage of the system. Value is $None$ if the system is in no alignment stage.
!+test stage+!	The current test mode stage of the system. Value is $None$ if the system is in no test stage.
!+new align stage+!	true while the alignment stage is changing. This includes an entry into an alignment stage from no alignment stage. This does not include entering no alignment stage.
!+new test stage+!	true while the test stage is changing. This includes an entry into a test stage from no test stage. This does not include entering no test stage.
!+CA stage complete+! !+CA2 stage complete+! !+CL stage complete+! !+CL2 stage complete+! !+ED stage complete+! !+ED2 stage complete+! !+FM stage complete+! !+FG stage complete+! !+HG stage complete+! !+HL stage complete+! !+HS stage complete+! !+ND stage complete+! !+ND2 stage complete+! !+TS stage complete+!	true iff the named alignment mode stage has been completed since entering the current alignment mode. Note that this does not preclude the possibility of the stage being re-entered before the completion of the mode.
!+AC stage complete+! !+CS stage complete+! !+DC stage complete+! !+DIO stage complete+! !+GA stage complete+! !+PD stage complete+! !+SC stage complete+! !+TM stage complete+!	true iff the named test mode has been completed since entering the current test mode. The stage may or may not be re-entered before the test mode is exited.

SS.STAGE.6 Undesired Event Dictionary: None.

SS.STAGE.7 System Generation Parameters: None

SS.STAGE.8 Information Hidden

1. The criteria for determining the current alignment stage.

2. What causes one stage to end and another to begin.

3. The number and order of stages in each of the alignment modes.

4. The criteria for determining the current test stage.

5. The number and order of test stages.

6. The fact that some alignment stages returned by this module are in fact leaves of a stage-hierarchy tree.

System Values Submodule

Introduction

A value returned by the System Values submodule has at least one of the following properties:

- it is used within one or more of the behavior-hiding modules;

- the rules governing its evaluation are shared or similar enough to another value returned by this module to warrant a shared implementation.

The rules for evaluation are the secret of this submodule. The submodule is divided into the following parts, based upon the semantic nature of the values returned:

(1) Device Reasonableness: These values are used to decide whether a device is providing reasonable results, based on device-independent criteria.

(2) IMS Alignment: These values are concerned with the alignment of the IMS platform. They include values which test the results of an IMS alignment process, as well as values dealing with the current state of the process.

(3) Reference Point: These values are based upon the position or attitude of the aircraft with respect to some point outside the aircraft. Some change as the aircraft's position changes (such as distances and bearings); others do not.

(4) Slew: These programs produce values concerning the slewing of symbols on a device, or the state of the slew control.

(5) Value Selection: Each value represents either (a) a choice among sensors or other modules to provide the value; or (b) a choice whether to pass on such a value at all, or to produce a null value for the variable. These values are the best available estimates of physical quantities, based upon the current status of the avionics system.

(6) Weapon Release: These are values involved with releasing a weapon. Some represent the relationship between the current a/c situation and a weapon release point. Others provide the status of certain aspects of the weapon release system.

System Values Module / Device Reasonableness Submodule

SS.SYSVAL.DEVREAS.1 Introduction

This submodule produces values used to decide whether a device is operating correctly and providing reasonable output, based on device independent criteria.

SS.SYSVAL.DEVREAS.2. Interface Overview

SS.SYSVAL.DEVREAS.2.1 Access Program Table

Program name	Parm type	Parm info	UEs
+G_ADC_ALT_UP+	pl:boolean;O	!+adc alt up+!	None.
+G_ADC_REASONABLE+	pl:boolean;O	!+adc reasonable+!	
+G_ADC_TAS_UP+	pl:boolean;O	!+adc tas up+!	
+G_DOPPLER_REASONABLE	pl:boolean;O	!+Doppler reasonable+!	
+G_DOPPLER_UP+	pl:boolean;O	!+Doppler up+!	
+G_IMS_REASONABLE+	pl:boolean;O	!+IMS reasonable+!	
+G_MACH_REASONABLE+	pl:boolean;O	!+mach reasonable+!	
+G_RADALT_REASONABLE+	pl:boolean;O	!+radalt reasonable+!	
+G_SR_REASONABLE+	pl:boolean;O	!+sr reasonable+!	

SS.SYSVAL.DEVREAS.2.2 Events Signalled

@T(!+adc reasonable+!)	@F(!+adc reasonable+!)
@T(!+adc tas up+!)	@F(!+adc tas up+!)
@T(!+Doppler up+!)	@F(!+Doppler up+!)
@T(!+IMS reasonable+!)	@F(!+IMS reasonable+!)
@T(!+SINS up+!)	@F(!+SINS up+!)
@T(!+sr reasonable+!)	@F(!+sr reasonable+!)

SS.SYSVAL.DEVREAS.3 Basic Assumptions

1. There are device-independent criteria for deciding the reasonableness of sensor inputs. The methods include checking for change over a period of time and insuring that the values fall within a specified range.

SS.SYSVAL.DEVREAS.4 Local Data Types: None.

SS.SYSVAL.DEVREAS.5 Dictionary

!+adc alt up+!	true iff the Air Data Computer is functioning and producing current and reasonable altitude readings.
!+adc reasonable+!	true iff the Air Data Computer is producing reasonable results for at least some of its reported values.
!+adc tas up+!	true iff the Air Data Computer is functioning and producing current and reasonable true airspeed readings.
!+Doppler reasonable+!	true iff the Doppler radar is producing reasonable groundspeed and drift angle readings.
!+Doppler up+!	true iff !+Doppler reasonable+!, and the groundspeed and drift angle readings produced by the Doppler are current.
!+IMS reasonable+!	true iff the IMS is giving reasonable results.
!+mach reasonable+!	true iff the current ADC mach reading is reasonable.
!+radalt reasonable+!	true iff the a/c is in an attitude when a reliable radar altimeter reading may be taken, and the current reading shows a reasonable value.
!+sr reasonable+!	true iff the a/c attitude and FLR configuration are such that reliable FLR slant range readings may be taken, and the FLR slant range value is reasonable.

SS.SYSVAL.DEVREAS.6 Undesired Event Dictionary: None.

SS.SYSVAL.DEVREAS.7 System Generation Parameters: None.

SS.SYSVAL.DEVREAS.8 Information Hidden

1. The criteria to judge validity or reasonableness of sensor inputs.

2. When and how often the reasonableness tests are performed.

System Values Module / IMS Alignment Submodule

SS.SYSVAL.IMSALN.1 Introduction

This submodule performs alignment tests of the IMS platform, and reports the results. It also provides other values concerned with the alignment of the IMS platform.

SS.SYSVAL.IMSALN.2. Interface Overview

SS.SYSVAL.IMSALN.2.1 Access Program Table

Program name	Parm type	Parm info	UEs
+G_ELAPSED_NAVALN_TIME+	p1:time;O	!+elapsed navaln time+!	None.

SS.SYSVAL.IMSALN.2.2 Events Signalled

@T(!+air velocity test passed+!)
@T(!+drift test failed+!)
@T(!+drift test passed+!)
@T(!+land velocity test failed+!)
@T(!+land velocity test passed+!)
@T(!+nav velocity test failed+!)
@T(!+SINS velocity test passed+!)

SS.SYSVAL.IMSALN.3 Basic Assumptions

1. It is possible to measure the elapsed time of certain phases of the alignment or navigation process. There is a maximum amount of time that will be measured; its value is determined by requirements, and set at system-generation time. If the elapsed time interval exceeds that maximum, the measurement will revert to zero and re-start.

2. There are certain tests to check the progress or results of an IMS platform alignment process. These tests compare IMS velocities with velocities obtained from other sources. If the velocities are within a particular threshold of each other, the test is considered to have passed; otherwise, it has failed. This module performs these tests, and reports their results. Only success or failure need be reported; other information is irrelevant to users.

SS.SYSVAL.IMSALN.4 Local Data Types: None.

SS.SYSVAL.IMSALN.5 Dictionary

!+air velocity test passed+!	true iff the difference between Doppler- and IMS-measured velocities is within acceptable bounds.
!+drift test failed+!	true iff the difference between the last value of !+gyro drift delta n+! and the current value of !+gyro drift delta n+! is too great.
!+drift test passed+!	true iff the difference between the last value of !+gyro drift delta n+! and the current value of !+gyro drift delta n+! is small enough.
!+elapsed navaln time+!	The elapsed time for which certain phases of alignment or navigation have proceeded, modulo the reset value, at which the measurement reverts to zero and resumes.
!+land velocity test failed+!	true iff the the last land velocity test is considered to have failed. The land velocity test is a test performed to determine the reliability of the IMS velocity measurements while the a/c is not airborne.
!+land velocity test passed+!	true iff the last land velocity test is considered to have been passed. The land velocity test is a test performed to determine the reliability of the IMS velocity measurements while the a/c is not airborne.

!+nav velocity test failed+!	true iff the differences between the Doppler- and IMS-measured velocities are not within acceptable bounds. Note that this is not the opposite of !+Air velocity test passed+!, as the acceptable bounds may differ in the two cases.
!+SINS velocity test passed+!	true iff the IMS-measured velocities are close enough to the SINS-measured velocities when compared.

SS.SYSVAL.IMSALN.6 Undesired Event Dictionary: None.

SS.SYSVAL.IMSALN.7 System Generation Parameters

#navaln_wraparound#	The maximum navigation/alignment time interval measurable by this module.

SS.SYSVAL.IMSALN.8 Information Hidden

1. When the navigation/alignment timer is started, stopped, and reset.
2. When the alignment tests are performed and the criteria defining success or failure.

System Values Module / Reference Point Submodule

SS.SYSVAL.REFPT.1 Introduction

The values produced by this submodule deal with reference points outside the aircraft. Some values (such as distances and bearings) are based upon the position or attitude of the aircraft with respect to some point, and these values change as the aircraft's position changes. Other values deal with locations of the points, and do not change as the aircraft moves.

SS.SYSVAL.REFPT.2 Interface Overview

SS.SYSVAL.REFPT.2.1 Access Program Table

Program name	Parm type	Parm info	UEs
	AIRCRAFT-POSITION-RELATIVE VALUES		
+G_BRG_AC_FTPT+	pl:angle;O	!+brg_ac_ftpt+!	None
+G_BRG_AC_TGT+	pl:angle;O	!+brg_ac_tgt+!	
+G_BRG_GRTK_AP+	pl:angle;O	!+brg_grtk_ap+!	
+G_BRG_GRTK_CUP+	pl:angle;O	!+brg_grtk_cup+!	
+G_BRG_GRTK_FTPT+	pl:angle;O	!+brg_grtk_ftpt+!	
+G_BRG_GRTK_OAP+	pl:angle;O	!+brg_grtk_oap+!	
+G_BRG_GRTK_TGT+	pl:angle;O	!+brg_grtk_tgt+!	
+G_GR_AC_FTPT+	pl:distance;O	!+gr_ac_ftpt+!	
+G_GR_AC_FXPT+	pl:distance;O	!+gr_ac_fxpt+!	
+G_GR_AC_HUDREFPT+	pl:distance;O	!+gr_ac_HUDrefpt+!	
+G_GR_AC_OAP+	pl:distance;O	!+gr_ac_oap+!	
+G_GR_AC_TGT+	pl:distance;O	!+gr_ac_tgt+!	
+G_HUDREFPT_AZ+	pl:angle;O	!+HUDrefpt_az+!	
+G_HUDREFPT_ELEV+	pl:angle;O	!+HUDrefpt_elev+!	
+G_LATITUDE+	pl:latitude;O	!+latitude+!	
+G_LATITUDE_ERROR+	pl:latitude;O	!+latitude error+!	
+G_LONGITUDE+	pl:longitude;O	!+longitude+!	
+G_LONGITUDE_ERROR+	pl:longitude;O	!+longitude error+!	
+G_SR_AC_AP+	pl:distance;O	!+sr_ac_ap+!	
+G_SR_AC_CUP+	pl:distance;O	!+sr_ac_cup+!	
+G_SR_AC_FTPT+	pl:distance;O	!+sr_ac_ftpt+!	
+G_SR_AC_GPUP+	pl:distance;O	!+sr_ac_gpup+!	
+G_SR_AC_OAP+	pl:distance;O	!+sr_ac_oap+!	
+G_SR_AC_TGT+	pl:distance;O	!+sr_ac_tgt+	
+G_STEERING_ERROR_TO_RLS+	pl:angle;O	!+steering error to rls+!	
+G_STEERING_ERROR_TO_TGT+	pl:angle;O	!+steering error to tgt+!	
+G_TIME_TO_FTPT+	pl:time;O	!+time to ftpt+!	

Program name	Parm type	Parm info	UEs
		FIXED-LOCATION VALUES	
+G_DEST_LAT+	p1:integer;I	locn nbr	%locn nbr
	p2:latitude;O	!+dest lat+!	illegal%
+G_DEST_LONG+	p1:integer;I	locn nbr	
	p2:longitude;O	!+dest long+!	
+G_MARK_LAT+	p1:integer;I	locn nbr	
	p2:latitude;O	!+mark lat+!	
+G_MARK_LONG+	p1:integer;I	locn nbr	
	p2:longitude;O	!+mark long+!	
+G_DESIG+	p1:boolean;O	!+desig+!	None
+G_LATITUDE_CUP+	p1:latitude;O	!+latitude_cup+!	
+G_LONGITUDE_CUP+	p1:longitude;O	!+longitude_cup+!	
+G_MARK+	p1:integer;O	!+mark+!	

SS.SYSVAL.REFPT.2.2 Events Signalled

```
@T(!+ap ahead+!)                         @F(!+ap ahead+!)
@T(!+cup ahead+!)                        @F(!+cup ahead+!)
@T(!+ftpt ahead+!)                       @F(!+ftpt ahead+!)
@T(!+oap ahead+!)                        @F(!+oap ahead+!)
@T(!+tgt ahead+!)                        @F(!+tgt ahead+!)

@T(!+desig+!)                            @F(!+desig+!)

@T(!+ground danger+!)                    @F(!+ground danger+!)

@T(!+gr_ac_ftpt+! lseq  20 nmi)
@T(!+gr_ac_ftpt+! gteq 10 nmi)           @F(!+gr_ac_ftpt+! gteq 10 nmi)
@T(!+gr_ac_ftpt+! gteq 1000 nmi)         @F(!+gr_ac_ftpt+! gteq 1000 nmi)
@T(!+gr_ac_ftpt+! lseq 30 nmi)           @F(!+gr_ac_ftpt+! lseq 30 nmi)
@T(!+gr_ac_fxpt+! lseq  22 nmi)
@T(!+gr_ac_HUDrefpt+! lseq 20 nmi)       @F(!+gr_ac_HUDrefpt+! lseq 20 nmi)
@T(!+gr_ac_HUDrefpt+! lseq 22 nmi)       @F(!+gr_ac_HUDrefpt+! lseq 22 nmi)
@T(!+gr_ac_HUDrefpt+! lseq 30 nmi)       @F(!+gr_ac_HUDrefpt+! lseq 30 nmi)
@T(!+gr_ac_HUDrefpt+! lseq 42 nmi)       @F(!+gr_ac_HUDrefpt+! lseq 42 nmi)
@T(!+gr_ac_oap+! gteq 10 nmi)            @F(!+gr_ac_oap+! gteq 10 nmi)
@T(!+gr_ac_oap+! gteq 1000 nmi)          @F(!+gr_ac_oap+! gteq 1000 nmi)
@T(!+gr_ac_oap+! lseq 30 nmi)
@T(!+gr_ac_tgt+! gteq 10 nmi)            @F(!+gr_ac_tgt+! gteq 10 nmi)
@T(!+gr_ac_tgt+! gteq 1000 nmi)          @F(!+gr_ac_tgt+! gteq 1000 nmi)
@T(!+gr_ac_tgt+! lseq 30 nmi)            @F(!+gr_ac_tgt+! lseq 30 nmi)
@T(!+gr_ac_tgt+! lseq 5000 ft)

@T( ABS(!+steering error to tgt+!) lseq 20 deg )
```

SS.SYSVAL.REFPT.3.1 Basic Assumptions

1. It is always possible to determine the distance and bearing from the aircraft to a specified point.

2. It is always possible to determine the azimuth and elevation relative to the aircraft Ya axis of a point on the ground, given the location of the point.

3. It is always possible to determine the aircraft's current position and altitude. Sometimes more than one estimate of each value are available.

4. The pilot can designate a non-moving point on the ground by positioning display symbols over that point (such as a HUD symbol), or the representation of that point on a display (such as cursors on the FLR). In the case of the map, the pilot can designate a point on the ground by causing the map to display that point. Once the display symbology indicates the desired point, the pilot takes an action to indicate designation. That point is known as the fix point.

5. The pilot can designate a non-moving point on the ground by positioning a HUD symbol over that point and then taking some action to indicate designation. That action is different than that taken to designate a fix point. This point is known as the adjusted point.

5. The pilot can designate one of a number of stored non-moving ground locations by setting certain controls. That designated location is known as the fly-to point.

6. The pilot can cause the coordinates of a geographic location to be displayed on the panel. That location is known as the called-up point.

7. The target is the place on the ground to which or over which the pilot wants to deliver a weapon or weapons. It may be identified to the system in several ways. Only one target may be defined at a time, although there may be many potential target locations stored. Targets are not necessarily stationary.

8. The offset aim point (oap) is a non-moving point on the ground near the target that the system may initially treat as the target (by displaying fly-to cues for it, or using its location in weapon delivery calculations, for example). When the aircraft gets near the offset aim point, the system will abandon it and use the real target location for subsequent calculations. The location of the oap may be designated to the system in one of several ways. There may only be one oap at any time, although there may be several potential oap locations stored.

9. In some modes, the system will define a point on the ground known as the HUD reference point. The system will cause certain HUD symbols to overlay this point, or may use its location in navigation or alignment activities. Sometimes the HUD reference point is fixed on the ground; sometimes it moves as the aircraft's position changes.

10. It is possible to store and recall the coordinates for a set of geographic points known as destinations. It is possible to store and recall the coordinates for a (possibly different) set of geographic points known as mark locations. Which set of destination or mark location coordinates is recalled is be determined by switch settings, by panel entries, or by conditions external to this module but which are accessible by this module.

11. Some designated ground points described in previous assumptions are not always defined. If a user requests information about a point that is currently undefined, the access program will return the last defined value, or a system-generation initial value if no other value has yet been defined.

SS.SYSVAL.REFPT.3.2 Assumptions about Undesired Events

1. User programs will not attempt to access a mark location or destination with a parameter that is out of bounds. The bounds are fixed at system generation time.

SS.SYSVAL.REFPT.4 Local Data Types: None.

SS.SYSVAL.REFPT.5 Dictionary

	true if and only if the
!+ap ahead+!	adjusted point
!+cup ahead+!	called-up point
!+ftpt ahead+!	fly-to point
!+oap ahead+!	offset aim point
!+tgt ahead+!	target
	is ahead of the aircraft; that is, iff the projection into the Xa-Ya plane of the line from the aircraft to the point has a positive Ya component.
!+brg_ac_ftpt+! !+brg_ac_tgt+!	The angle measured clockwise (looking down) from the projection into the horizontal plane of the aircraft's Ya axis to the projection into the horizontal plane of the line from the aircraft to the fly-to point and the target, respectively.

	The bearing measured clockwise (looking down) from the projection into the horizontal plane of the aircraft's ground track to the projection into the horizontal plane of the line from the aircraft to:
!+brg_grtk_ap+!	the adjusted point;
!+brg_grtk_cup+!	the called-up point;
!+brg_grtk_ftpt+!	the fly-to point;
!+brg_grtk_oap+!	the offset aim point;
!+brg_grtk_tgt+!	the target.
!+desig+!	true iff a reference point outside the a/c has been designated.
!+dest lat+!	the latitude of destination pl.
!+dest long+!	the longitude of destination pl.
	The ground range from the aircraft's present position to:
!+gr_ac_ftpt+!	the fly-to point;
!+gr_ac_fxpt+!	the fix point;
!+gr_ac_HUDrefpt+!	the HUD reference point;
!+gr_ac_oap+!	the offset aim point;
!+gr_ac_tgt+!	the target.
!+ground danger+!	true iff the pilot must execute an immediate 4g pullup to avoid striking the ground.
!+HUDrefpt_az+!	The angle between the Ya axis, and the projection into the Xa-Ya plane of the ray from the aircraft to the current HUD reference point. The angle is positive if the ray is to the right (looking down) of the Ya axis.
!+HUDrefpt_elev+!	The angle between the Ya axis, and the projection into the Ya-Za plane of the ray from the aircraft to the current HUD reference point. The angle is positive if the ray is above (positive Za direction) the plane.

!+latitude+!	The present latitude of the aircraft.
!+latitude_cup+!	The latitude of the called-up point.
!+latitude error+!	The difference in latitude between the two current positional reference points.
!+longitude+!	The present longitude of the aircraft.
!+longitude_cup+!	The longitude of the called-up point.
!+longitude error+!	The difference in longitude between the two current positional reference points.
!+mark+!	The number associated with the most recently defined Mark destination. Ranges from 1 to #num_mark_locns#.
!+mark lat+!	The latitude of mark position p1.
!+mark long+!	The longitude of mark position p1.
	The slant range from the aircraft to:
!+sr_ac_ap+!	the adjusted point;
!+sr_ac_cup+!	the called-up point;
!+sr_ac_ftpt+!	the fly-to point;
!+sr_ac_oap+!	the offset aim point;
!+sr_ac_tgt+!	the target.
!+sr_ac_gpup+!	The slant range (straight-line) distance to the point where @T(!+ground danger+!) will occur due to ground proximity should the a/c continue its present course.
!+steering error to rls+!	The angle between the a/c ground track and the horizontal line from the a/c to the point where the a/c should release the current weapon in order to strike the target. Positive if the ground track line is to the left of the line to the release point (looking down).

!+steering error to tgt+!	The angle between the a/c ground track and the horizontal line from the a/c to the target. Positive if the ground track line is to the left of the line to the target (looking down). This measures the same angle as !+brg_grtk_tgt+!, except it is measured not on a full-circle scale, but rather on a plus-minus angle basis.
!+time to ftpt+!	The time to go before the a/c reaches the fly-to point, assuming a direct flight toward the point at the current horizontal velocity. Always positive.

SS.SYSVAL.REFPT.6 Undesired Event Dictionary

%locn nbr illegal%	A user program requested the latitude or longitude of a mark location or destination greater than #num_dests# or less than 1.

SS.SYSVAL.REFPT.7 System Generation Parameters

#locn_init_lat# #locn_init_long#	The initial latitude and longitude of the called-up point, the fly-to point, the target, the offset aim point, the fix point, the adjusted point, the HUD reference point, all mark locations, and all destination locations.
#num_dests#	The maximum number of sets of destination coordinates that may be recalled.
#num_mark_locns#	The maximum number of sets of Mark coordinates that may be recalled.

SS.SYSVAL.REFPT.8 Information Hidden

1. The definition of the HUD reference point, fly-to point, called-up point, target, offset aim point, adjusted point, and fix point; what actions are taken to designate the reference points; what device determines the fix point under what conditions.

2. The latitude/longitude error reference points.

3. For ground and slant ranges, which of the available earth terrain models is used for the calculations.

4. How the value of !+mark+! is determined.

5. Where the destination and mark locations come from; when they are updated and with what values.

System Values Module / Slew Submodule

SS.SYSVAL.SLEW.1 Introduction

This module produces values concerning the way in which the slew control is used to change the position of certain symbols and displays. Some values are displacements (angular or linear), to tell how far to move a symbol that is being slewed. Other values give information about the state of the slewing process.

SS.SYSVAL.SLEW.2 Interface Overview

SS.SYSVAL.SLEW.2.1 Access Program Table

Program name	Parm type	Parm info	UEs
+G_AFTER_SLEWING+	p1:boolean;O	!+after slewing+!	None.
+G_BEFORE_SLEWING+	p1:boolean;O	!+before slewing+!	
+G_DURING_SLEWING+	p1:boolean;O	!+during slewing+!	
+G_SLEW_FLR_DISPL+	p1:angle;O	!+slew FLR delta az+!	
	p2:distance;O	!+slew FLR delta rng+!	
+G_SLEW_HUD_DISPL+	p1:angle;O	!+slew HUD delta az+!	
	p2:angle;O	!+slew HUD delta elev+!	
+G_SLEW_MAP_DISPL+	p1:latitude;O	!+slew map delta lat+!	
	p2:longitude;O	!+slew map delta long+!	
+G_HUD_SLEW_LEGAL+	p1:boolean;O	!+HUD slew legal+!	

SS.SYSVAL.SLEW.2.2 Events Signalled: None

SS.SYSVAL.SLEW.3 Basic Assumptions

1. Certain symbols and displays are moved under software control as the result of the pilot manipulating the slew control. These include the HUD aiming symbol, FLR cursors, and the map display.

2. For those symbols/displays which are slewed, the positions are updated periodically, with constant period. How much a symbol/display should be displaced from its last position is a function of the desired rate of motion, and how often the symbol/display position is updated.

3. This module assumes that the position of a symbol/display being slewed will be updated fast enough to simulate continuous motion. That update rate is available from another module.

4. The desired rate of motion for each symbol/display (known as slew rate) is defined by requirements. There are three slew rates, one for each symbol/display that can be slewed. The HUD aiming symbol will be slewed at the HUD rate; FLR symbols will be slewed at the radar rate; the map will be slewed at the map rate.

5. Slew control displacement is available to this module in lateral and vertical components. Lateral displacement of the slew control should cause the symbol/display being slewed to have (only) a lateral component of motion. Vertical displacement of the slew control should cause the symbol/display being slewed to have (only) a vertical component of motion.

6. There are times when the HUD aiming symbol will not be moved as the result of inputs from the slew control. When this is the case is determined by requirements, and reported out by this module.

SS.SYSVAL.SLEW.4 Local Data Types: None.

SS.SYSVAL.SLEW.5 Dictionary

!+*after slewing*+!	true *iff the system is in a navigation* update or weapon delivery mode and legal slew inputs have been entered at least once since mode entry.
!+before slewing+!	true iff the system is in a navigation update or weapon delivery mode, and no legal slew inputs have yet been entered since mode entry.
!+during slewing+!	true iff the system is in a navigation update or weapon delivery mode, and a legal slewing operation is currently in progress.
!+HUD slew legal+!	true iff the HUD aiming symbol is allowed to be slewed. If not, then inputs from the slew control should have no effect on the symbol's position.

!+slew FLR delta az+!	How much a slewed FLR symbol should be shifted in azimuth, given the current slew control position. Positive value means right (as seen by the pilot).
!+slew FLR delta rng+!	How much a slewed FLR symbol should be shifted in range, given the current slew control position. Positive value means range location of symbol should be increased.
!+slew HUD delta az+!	How much a slewed HUD symbol should be shifted in azimuth, given the current slew control position. Positive value means right (as seen by the pilot).
!+slew HUD delta elev+!	How much a slewed HUD symbol should be shifted in elevation, given the current slew control position. Positive value means up (as seen by the pilot).
!+slew map delta lat+!	How much the map display should be shifted in latitude, given the current slew control position.
!+slew map delta long+!	How much the map display should be shifted in longitude, given the current slew control position.

SS.SYSVAL.SLEW.6 Undesired Event Dictionary: None.

SS.SYSVAL.SLEW.7 System Generation Parameters: None.

SS.SYSVAL.SLEW.8 Information Hidden

1. How the slew rates are computed, and how they are a function of the slew control displacement.
2. How the program decides if the current state is !+before slewing+!, !+after slewing+!, or !+during slewing+!. What constitutes a legal slew input.

System Values Module / Value Selection Submodule

SS.SYSVAL.VALSEL.1 Introduction

The Function Driver occasionally requires values that may be obtained from more than one source. This submodule provides such values by making the appropriate choices. It produces values provided directly by the virtual devices or computed directly by other system modules. Each value represents either (a) a choice among sources to provide the value; or (b) a choice whether to accept input from a source at all, or to return a null (zero) value.

SS.SYSVAL.VALSEL.2 Interface Overview

SS.SYSVAL.VALSEL.2.1 Access Program Table

Program name	Parm type	Parm info	UEs
+G_IN_FLIGHT+	p1:boolean;O	!+in_flight+!	None
+G_ALT_PRIORITY_DISPLAY+	p1:distance;O	!+alt priority stale+!	
	p2:sensor_name;O	!+alt priority source+!	
+G_ALT_PRIORITY_RANGING+	p1:distance;O	!+alt priority ranging+!	
+G_DRIFT_ANGLE+	p1:angle;O	!+drift angle+!	
+G_VELOCITY_EAST_SYSTEM+	p1:speed;O	!+velocity east system+!	
+G_VELOCITY_NORTH_SYSTEM+	p1:speed;O	!+velocity north system+!	
+G_VELOCITY_VERTICAL_SYSTEM+	p1:speed;O	!+velocity vertical system+!	
+G_WIND_DIR+	p1:angle;O	!+wind dir+!	
+G_WIND_VEL+	p1:speed;O	!+wind vel+!	

SS.SYSVAL.VALSEL.2.2 Events Signalled

@T(!+in flight+!) @F(!+in flight+!)

SS.SYSVAL.VALSEL.3 Basic Assumptions

1. Most measurements of the aircraft's situation or environment may be obtained from more than one source. There is usually a preferred source; the actual source may depend on sensor status and/or aircraft situation.

2. It is possible to tell when the aircraft is airborne.

SS.SYSVAL.VALSEL.4 Local Data Types

sensor_name	Enumerated: $None$, A, F, H.

SS.SYSVAL.VALSEL.5 Dictionary

!+alt priority stale+!	The value !+alt priority ranging+! frozen at some specified moment in the recent past. When the value is updated is determined by certain events and mode transitions during the flight.
!+alt priority ranging+!	The current altitude of the aircraft, from the best available sensors.
!+alt priority source+!	The sensor from which the current value of !+alt priority stale+! was obtained.
!+drift angle+!	The horizontal angle between aircraft heading and the aircraft horizontal velocity vector.
!+in_flight+!	true iff the a/c is airborne.
!+velocity east system+!	The aircraft's current east velocity component, computed from the best available sources.
!+velocity north system+!	The aircraft's current north velocity component, computed from the best available sources.
!+velocity vertical system+!	The aircraft's current vertical velocity component, computed from the best available sources.
!+wind dir+!	The direction that the wind is blowing from, measured on a circle scale with true North at the origin. Obtained from best available sources.
!+wind vel+!	The current wind speed, obtained from best available sources.

SS.SYSVAL.VALSEL.6 Undesired Event Dictionary: None.

SS.SYSVAL.VALSEL.7 System Generation Parameters: None.

SS.SYSVAL.VALSEL.8 Information Hidden

1. Which sensors provide the values produced.
2. How the choice among various sensors is made.
3. How the choice between providing a sensor value or providing a null value is made.
4. When various values are determined and/or updated.

System Values Module / Weapon Release Submodule

SS.SYSVAL.WPNRLS.1 Introduction

This submodule produces values that represent the relationship between the current aircraft situation and a weapon release.

SS.SYSVAL.WPNRLS.2 Interface Overview

SS.SYSVAL.WPNRLS.2.1 Access Program Table

Program name	Parm type	Parm info	UEs
+G_AZ_MISS_DIST+	p1:distance;O	!+az miss dist+!	%no wpn mode%
+G_GR_AC_RMAX+	p1:distance;O	!+gr_ac_rmax+!	
+G_HIGH_DRAG_RELEASE+	p1:boolean;O	!+high drag release+!	
+G_LOW_DRAG_RELEASE+	p1:boolean;O	!+low drag release+!	
+G_RLS_PTS_PASSED+	p1:integer;O	!+rls_pts_passed+!	
+G_SR_AC_BTPUP+	p1:distance;O	!+sr_ac_btpup+!	
+G_SR_AC_IP+	p1:distance;O	!+sr_ac_ip+!	
+G_SR_AC_RLS+	p1:distance;O	!+sr_ac_rls+!	
+G_STIK_QUAN+	p1:integer;O	!+stik_quan+!	
+G_WPNS_RLSD+	p1:integer;O	!+wpns rlsd+!	
+G_GAS+	p1:boolean;O	!+GAS+!	
+G_OTS+	p1:boolean;O	!+OTS+!	
+G_STEERING_TO_TGT+	p1:boolean;O	!+steering to tgt+!	
+G_TOS+	p1:boolean;O	!+TOS+!	

SS.SYSVAL.WPNRLS.2.2 Events Signalled

```
@T(!+blast danger+!)              @F(!+blast danger+!)
@T(!+computed rls+!)
@T(!+GAS+!)                       @F(!+GAS+!)
@T(!+high drag release+!)         @F(!+high drag release+!)
@T(!+low drag release+!)          @F(!+low drag release+!)
@T(!+OTS+!)                       @F(!+OTS+!)
@T(!+rls_pts_passed+! = 1)
@T(!+r65+!)                       @F(!+r65+!)
@T(!+rmax+6000+!)                 @F(!+rmax+6000+!)
@T(!+rmin+!)                      @F(!+rmin+!)
@T(!+rmin+6000+!)                 @F(!+rmin+6000+!)
@T(!+special in range+!)          @F(!+special in range+!)
@T(!+special solution+!)          @F(!+special solution+!)
@T(!+steering to tgt+!)           @F(!+steering to tgt+!)
@T(!+stik created+!)
@T(!+stik empty+!)
@T(!+target in range+!)           @F(!+target in range+!)
@T(!+time to prepare+!)
@T(!+TOS+!)                       @F(!+TOS+!)
```

SS.SYSVAL.WPNRLS.3.1 Basic Assumptions

1. A release point is a point in space such that if the current weapon were released at that point (assuming current aircraft velocity and attitude), it would successfully strike within an acceptable neighborhood of the target. It is possible to determine the distance between the a/c and a release point. It is possible to determine the distance between the a/c and an impact point.

2. A weapon will not necessarily be released when a release point is encountered. To effect an actual release, certain pilot actions are necessary as the aircraft passes through the release point.

3. The pilot is able to affect the delivery characteristics of certain kinds of weapons. He may choose between states known as high drag and low drag. If such an alterable weapon is selected, then either high drag or low drag will be true, but not both. If such a weapon is not chosen (or some un-alterable weapon type is chosen), then both conditions will be reported out as false.

4. Sometimes weapons are delivered in fixed numbers called stiks. The weapons in a stik are released consecutively, not simultaneously. A stik has at least one weapon in it. All weapons in a stik are of the same weapon class. There are some weapon classes for which stik deliveries are not allowed. This module can determine if the next weapon delivery will be a stik delivery.

5. For each weapon in a stik there is a corresponding computed release point in the air and impact point on the ground. The distance between each impact point is the same for any one stik, although that distance may vary between stiks. The number of weapons in a stik and the delivery spacing between individual weapons in a stik are functions of pilot selection and weapon class, limited by requirements. If the pilot selection changes during the delivery of a stik, the change will be ignored and the stik delivered as though the change had not been entered.

6. This module can determine the number of release points in the current stik that have already passed, and the number of weapons in that stik that have actually been released.

7. It is hazardous for the aircraft to come within a certain distance of the blast effects of a weapon that it has delivered. This module reports how far away the aircraft is from the danger area, and signals when the aircraft has crossed into it.

8. In weapon delivery modes, there are four steering stages. They are known as go-around, over-the-shoulder, steering-to-target, and tail-on, respectively. Each one represents a relationship between the aircraft and its target, and is used to determine how the aircraft should be flown in order to carry out weapon delivery. The system may be in at most one steering stage at a time.

SS.SYSVAL.WPNRLS.3.2 <u>Assumptions about Undesired Events</u>

1. No value produced by this module has any meaning unless the system is in some weapon delivery mode. User programs will not call an access program in this module if the system is not in a weapon delivery mode.

SS.SYSVAL.WPNRLS.4 <u>Local Data Types</u>: None.

SS.SYSVAL.WPNRLS.5 <u>Dictionary</u>

!+az miss dist+!	The distance along the ground between the target and the ground-projected line from the aircraft to the computed impact point.
!+blast danger+!	<u>true</u> iff the pilot should immediately execute a 4g pullup to avoid dangerous weapon blast effects.
!+computed rls+!	<u>true</u> iff the active weapon would strike the target, or within an acceptable neighborhood of the target, if it were released right now.
!+GAS+!	<u>true</u> iff the current steering state is go-around-steering.
!+gr_ac_rmax+!	The ground range between the aircraft's present position and the position where the condition !+Rmax+! will be true.
!+high drag release+!	<u>true</u> iff a weapon type with alterable delivery characteristics has been chosen, and the pilot has selected the high drag configuration.
!+low drag release+!	<u>true</u> iff a weapon type with alterable delivery characteristics has been chosen, and the pilot has selected the low drag configuration.
!+OTS+!	<u>true</u> iff the current steering state is over-the-shoulder steering.
!+rls_pts_passed+!	The number of computed release points for the latest stik that have already been passed.
!+r65+!	<u>true</u> iff the a/c is in the proper configuration to release a special weapon at a 65 degree flight path angle.

!+rmax+!	true iff the a/c is in the proper configuration and that the a/c is at a maximum pullup range that would result in a special weapon impacting the target
!+rmax+6000+!	true iff the a/c is in the proper configuration and that the a/c is at a pullup range that would result in a special weapon impacting 6000 feet short of the target.
!+rmin+!	true iff the a/c is in the proper configuration and that the a/c is at a minimum pullup range that would result in a special weapon impacting the target.
!+rmin+6000+!	true iff the a/c is in the proper configuration and that the a/c is at a pullup range that would result in a special weapon impacting 6000 feet long of the target.
!+special in range+!	true iff the target is in range for a special weapon.
!+special solution+!	true iff the current miss distance (i.e., how far the weapon will miss the target if released now) and steering error are both within acceptable bounds.
!+sr_ac_btpup+!	The slant range (straight-line) distance to the point where @T(!+blast danger+!) will occur due to weapon blast effects should the a/c continue its present course.
!+sr_ac_ip+!	The slant range from the a/c's present position to the computed impact point of the next weapon in the stik.
!+sr_ac_rls+!	The slant range (straight-line distance) from the a/c to the point where the next weapon release should occur.
!+steering to tgt+!	true iff the current steering state is steering-to-target.
!+stik created+!	The aircraft has not yet passed the first release point in this stik.

!+stik empty+!	The aircraft has passed the last release point in the current stik.
!+stik_quan+!	The possible number of release points for the current stik.
!+target in range+!	true iff the target is in range of the current weapon type.
!+time to prepare+!	true iff !+time to rls+! lseq !+preparation time+! for the current weapon.
!+TOS+!	true iff the current steering state is tail-on-steering.
!+wpns rlsd+!	The number of weapons in the current stik that have actually been released.

SS.SYSVAL.WPNRLS.6 Undesired Event Dictionary

%no wpn mode%	An access program was called when the system was in no weapon delivery mode.

SS.SYSVAL.WPNRLS.7 *System Generation Parameters:* None

SS.SYSVAL.WPNRLS.8 Information Hidden

1. How the number of weapons in a stik is determined. How the delivery spacing for a stik is determined. What classes of weapons cannot be delivered in a stik.
2. What configurations determine a high or low drag release condition, and what weapon types are applicable to the conditions.
3. How to tell if the conditions for a successful weapon release exist.
4. How to tell if a target is in range or not.
5. How to tell what weapon delivery steering stage the system is in (if any).
6. How to determine the radius for blast avoidance.

APPENDIX 1

Interface Design Issues

SS.STAGE

1. Some alignment stages in fact comprise two substages. Rather than impose some hierarchical structure on the interface (such as a GET SUBSTAGE program), we chose to treat the secondary substages as stages in their own right. Because there are at most two substages, we have represented the applicable cases as either stage "XX" or stage "XX2". The advantage is that this simplifies the interface.

2. Should it be a UE to ask for the current alignment stage if the system is in no alignment mode? We decided not, because there is no sure way to avoid generating that UE. The system could leave the alignment mode after the user program checked to be sure that one was in progress, but before the alignment stage request. By using $None$ values, we avoid making user programs perform the mode check.

3. The interface tells whether a certain stage is completed or not so that the order of the stages within a mode remains hidden.

SS.SUBRTN

1. Should it be an undesired event to attempt to use +SS_SYMBOL_AZ_ON_ASL+ when the HUD ASL is turned off? We decided not, because the HUD DIM provides the location of a HUD symbol regardless of that symbol's current mode.

SS.MODE

1. In a previous version of this module, users had no facilities for defining alias mode names.

   We decided that direct use of system modes made user programs unnecessarily vulnerable to changes in the mode module; i.e. a large class of likely changes to the mode module would result in changes wherever the affected modes had been used in system documentation and programs. Changes of this type include changing the functions performed in a mode, dividing the functions performed in a mode between two or more new modes, and adding/deleting a mode. By providing a facility for defining alias mode names, user programs are insulated from most classes of changes to the mode module. Adding or deleting a mode becomes simpler, especially where the Function Driver programs need not change. Changing the functions performed in a mode does not affect FD programs at all except where operations must be added or deleted. The division of a mode into two or more new modes becomes trivial. In most situations, changes to the mode module will only require that some alias mode definitions be changed.

2. We considered hiding the system modes completely within the MODE module and specifying the correspondence between system modes and alias modes as a secret of the module. This alternative has the advantage of making it unnecessary for user programs to know anything about the system modes or define any mode correspondences. It has the disadvantage of requiring substantial changes to the current documentation; further, the system modes are drawn from classic avionics terminology, and make for easier reviews. We decided to accept the first alternative.

SS.PNL.CONFIG

1. The panel may simultaneously be in two configurations because there are two windows. The requirements considers each upper-window/lower-window display function as <u>one</u> display using two windows. We consider the pairings arbitrary and changeable; therefore, we consider each function to be <u>two</u> displays in one window each. Having decided that, we must consider each state of the relevant switches to define <u>two</u> configurations.

2. Formerly, we provided an access program that returned one or both of the current configurations, rather than the current program that provides a yes-or-no answer to a user's "guess". We adopted the latter scheme because it is much simpler than its predecessor, and because it is all that user programs need.

SS.PNL.FORMAT

1. It was decided not to include any services dealing with the Mark window in this module. Because the Mark window is so simple, any such services would have duplicated what is already provided by the virtual device. Thus, to operate the Mark window, user programs should invoke the DIM access programs directly.

2. For the upper and lower window, there are services provided here which in fact duplicate services provided by the virtual device. (For instance, the effect of +SS_CLEAR_UPPER+ may be duplicated by calling +DI_S_UPPER_WINDOW+ with a blank argument.) If this was not done, user programs would have to use two different interfaces to drive the upper/lower display windows. It is intended that all user programs affecting the upper/lower display windows use this module <u>exclusively</u>.

3. We provided the blank-lights format (which turns on the format lights and blanks out the window) because we wanted to make sure that only one module (namely, this one) controls the DIM panel format lights. The alternative (which appeared in an earlier design) was to have user program control the DIM format light programs directly.

4. We disclosed information about each display format in the program effects section so that users could choose which display program meets their requirements.

SS.PNL.INPUT

1. The events @T(!+Input requested+!) and @T(!+Input attempted+!) are signalled because they are required by the Function Driver module that controls the panel enter light.

SS.SYSVAL.DEVREAS

1. Although !+Doppler up+! is not currently used by a function driver, it is used by more than one shared service module. Similarly, !+Doppler reasonable+! is only used by one function driver; however, it is also used by a shared services module. Therefore, both are included in this shared service module.

2. Some values are signalled as events, and others given only via access programs. This is done because we provide exactly what is needed by user modules.

SS.SYSVAL.IMSALN

1. The events signalled by this module are exactly those required by the Function Driver modules. Therefore, for some tests, only a failure is reported; for others, only a success; for still others, both can be signalled.

2. Currently, !+elapsed navaln time+! is only used by one function driver. However, we suspect that it may be used in the implementation of the SS mode-reporting module. Therefore, we have included it in this module, but are prepared to withdraw it at a later time.

SS.SYSVAL.REFPT

1. The latitude and longitude of the aircraft clearly deal with external reference points (namely, the aircraft's present position). However, the values are also based on a choice among sources (SINS, panel entry, PM calculations). Should those values go here, or in the sensor-choice submodule? We decided here, because it is more natural to think of positional information as being position-relative, rather than the result of a sensor choice. The existence of a source choice is in fact hidden by this submodule.

2. Some values produced by this module are only used by one function driver. However, they are used by other shared services modules, and so qualify for inclusion in this shared services module.

SS.SYSVAL.SLEW

1. A former version of this module provided rates, not displacements. That was because slewing used to be a builtin facility of the virtual map, aiming symbol, and flr cursors. When that facility was deleted from those devices (see the design issues in the device specifications), this module changed accordingly.

2. We considered having user programs pass in the slew control displacement as parameters to the symbol-displacement programs. We rejected this, however, because it would serve no useful purpose; and because deleting these parameters would simplify the interface.

3. We could have added conditions for the legality of FLR and map slews. The programs that control the FLR cursors and the map display would have to check for the legality of a slew before positioning the symbol. However, it is currently the case that the FLR and map can be slewed anytime they are displayed. Therefore, the controlling program would always get true for an answer. We decided not to add this needless condition. If requirements change so that the FLR cursors and the map cannot be slewed under some conditions, then we will have to add slew-legality programs for them.

4. We added the !+HUD slew legal+! value to this module because although used by only one Function Driver, it also affects the values of !+after slewing+! and !+before slewing+!.

SS.SYSVAL.VALSEL

None.

SS.SYSVAL.WPNRLS

1. Values such as !+target in range+! and !+blast danger+! and !+sr_ac_ip+! could have been included in the aircraft position-relative values, since their values change as the aircraft flies. However, since they also depend on weapon characteristics, and because we felt it was more natural to think of them in a weapon module, they were included here.

APPENDIX 2

Implementation Notes

Implementation Notes for each Shared Services submodule are found in this appendix. Entries corresponding to interface terms give the Requirements-based definition for the value, expressed in terms of the facilities of this or other modules. Entries in the form of local dictionary terms (bracketed by "!!") are definitions of values that are not reported out over the interface, but that are used in a definition of an interface term.

This appendix is divided into sections for each submodule, presented in Table of Contents order.

SS.PNL.CONFIG

1. !+Pnl config changed+! Signalled whenever @T(!+Map hold changed+!) OR @T(!+Update changed+!) OR @T(!+Panel mode changed+!) OR @T(!+Pres pos changed+!) occurs.

2. The requirements definitions of each possible panel configuration are enumerated below.

Nearly all of the configuration values are functions of the !+Map hold+!, !+Update+!, !+Panel mode+!, and !+Pres pos+! switch values. In addition, some are functions of pilot-entered keyboard input. In the table that follows, the appropriate values of the switches are given; and "X" means that particular switch value does not affect the value of the item. Under the "Kbd. input" column, an "X" means that no keyboard input is needed to define the item. A "D" means that a destination must be selected; i.e., @T(!+Dest selected+!) must occur before the item is considered true. If a two-digit data selection number is required, its value (or legal range) is given. Note that if any keyboard input is called for, it must be entered <u>after</u> the switches have taken on their assigned values; however, the order of the switch settings is immaterial.

The panel may be in at most two configurations at once.

Table SS.PNL.CONFIG.12-a
Definition of !+pnl config+!

	!+Map hold+!	!+Update+!	!+Panel mode+!	!+Pres pos+!	Kbd. input
$align_stage$	<u>false</u>	$Data$	$Prespos$	$Update$	88
$alt AGL at rls$	<u>false</u>	$Data$	$Prespos$	$Update$	81
$alt baro AGL$	<u>false</u>	$Data$	$Prespos$	$Update$	80
$ARPINT$	<u>false</u>	$Data$	$Prespos$	$Update$	84
$ARPQUANT$	<u>false</u>	$Data$	$Prespos$	$Update$	84
$az miss dist at rls$	<u>false</u>	$Data$	$Prespos$	$Update$	82
$az ref hdg$	<u>false</u>	$IMS-HUD$	$Prespos$	$Update$	X
$burst ht$	<u>false</u>	X	$DBHT$	X	D
$central long a$	<u>false</u>	$Data$	$Prespos$	$Update$	17
$central long b$	<u>false</u>	$Data$	$Prespos$	$Update$	18
$data nbr$	<u>false</u>	$Data$	$Prespos$	$Update$	00-26
$dest altitude$	<u>false</u>	X	$ALTMSLP$	X	D
$dest lat$	<u>false</u>	X	$Dest$	X	D

SS.PNL.CONFIG (continued)

	!+Map	!+Update+!	!+Panel	!+Pres	Kbd.
	hold+!		mode+!	pos+!	input
$dest long$	false	X	$Dest$	X	D
$dest mslp$	false	X	$ALTMSLP$	X	D
$Doppler coupled$	false	$Data$	$Prespos$	$Update$	25
$drift angle filtered$	false	$Data$	$Prespos$	$Update$	96
$drftangl IMS$	false	$Data$	$Prespos$	$Update$	96
$e coarse scale$	false	$Data$	$Prespos$	$Update$	08
$e fine scale$	false	$Data$	$Prespos$	$Update$	11
$e coarse bias$	false	$Data$	$Prespos$	$Update$	13
$e fine bias$	false	$Data$	$Prespos$	$Update$	16
$elapsed navaln time$	false	$Data$	$Prespos$	$Update$	88
$fpangl at rls$	false	$Data$	$Prespos$	$Update$	82
$gndspd filtered$	false	$Data$	$Prespos$	$Update$	95
$groundspeed IMS$	false	$Data$	$Prespos$	$Update$	95
$gyro drift delta n$	false	$Data$	$Prespos$	$Update$	00
$hdg system$	false	$Data$	$Prespos$	$Update$	91
$heading IMS$	false	$IMS-HUD$	$Prespos$	$Update$	X
OR	false	$Data$	$Prespos$	$Update$	73
$heading MAG$	false	$Data$	$Prespos$	$Update$	73
$IMS diags1$	false	$Da[illegible]$	$Prespos$	$Update$	72
$IMS diags2$	false	$Data$	$Prespos$	$Update$	72
$IMS total vel$	false	$Data$	$Prespos$	$Update$	97
$L-probe$	false	$Data$	$Prespos$	$Update$	26
$land based$	false	$Data$	$Prespos$	$Update$	23
$latitude$	false	X	$Prespos$	$LatLong$	X
$longitude$	false	X	$Prespos$	$Latlong$	X
$low lat ct a$	false	$Data$	$Prespos$	$Update$	19
$low lat ct b$	false	$Data$	$Prespos$	$Update$	20
$mag var$	false	$Tacmv$	$Prespos$	$Update$	D
$map latitude$	true	X	X	X	X
$map longitude$	true	X	X	X	X
$map orient a$	false	$Data$	$Prespos$	$Update$	21
$map orient b$	false	$Data$	$Prespos$	$Update$	22
$map sw diags$	false	$Data$	$Prespos$	$Update$	71
$MFSW diags$	false	$Data$	$Prespos$	$Update$	71
$mark lat$	false	X	$Mark$	X	D
$mark long$	false	X	$Mark$	X	D
$n coarse scale$	false	$Data$	$Prespos$	$Update$	07
$n fine scale$	false	$Data$	$Prespos$	$Update$	10
$n coarse bias$	false	$Data$	$Prespos$	$Update$	12
$n fine bias$	false	$Data$	$Prespos$	$Update$	15

SS.PNL.CONFIG (continued)

	!+Map	!+Update+!	!+Panel	!+Pres	Kbd.
	hold+!		mode+!	pos+!	input
$nav diags1$	false	$Data$	$Prespos$	$Update$	98
$nav diags2$	false	$Data$	$Prespos$	$Update$	98
$norm accel at rls$	false	$Data$	$Prespos$	$Update$	83
$offset brg$	false	X	Rng/Brg	X	D
$offset dht$	false	X	$DBHT$	X	D
$offset rng$	false	X	Rng/Brg	X	D
$OFP ver1$	false	$Data$	$Prespos$	$Update$	99
$OFP ver2$	false	$Data$	$Prespos$	$Update$	99
$priority alt display$	false	$Data$	$Prespos$	$Update$	80
$radalt priority$	false	$Data$	$Prespos$	$Update$	24
$SINS dhdg$	false	$Z-DHDG$	$Prespos$	$Update$	X
$SINS east vel$	false	$Data$	$Prespos$	$Update$	93
$SINS heading$	false	$Data$	$Prespos$	$Update$	91
$SINS lat$	false	$Data$	$Prespos$	$Update$	90
$SINS long$	false	$Data$	$Prespos$	$Update$	90
$SINS north vel$	false	$Data$	$Prespos$	$Update$	92
$SINS valid1$	false	$Data$	$Prespos$	$Update$	94
$SINS valid2$	false	$Data$	$Prespos$	$Update$	94
$SINS x offset$	false	$SINSX-Y$	$Prespos$	$Update$	X
$SINS y offset$	false	$SINSX-Y$	$Prespos$	$Update$	X
$SINS z offset$	false	$Z-DHDG$	$Prespos$	$Update$	X
$slant range at rls$	false	$Data$	$Prespos$	$Update$	81
$TAS ADC at rls$	false	$Data$	$Prespos$	$Update$	83
$TAS filtered$	false	$Data$	$Prespos$	$Update$	97
$time to dest$	false	$Data$	$Prespos$	$Update$	89
$STARDY diags$	false	$Data$	$Prespos$	$Update$	70
$v coarse scale$	false	$Data$	$Prespos$	$Update$	09
$v coarse bias$	false	$Data$	$Prespos$	$Update$	14
$vel e$	false	$Data$	$Prespos$	$Update$	93
$vel n$	false	$Data$	$Prespos$	$Update$	92
$wpn sw diags$	false	$Data$	$Prespos$	$Update$	70
$WEAPTYP$	false	$Data$	$Prespos$	$Update$	85
$wind dir$	false	X	$Prespos$	$Wind$	X
$wind vel$	false	X	$Prespos$	$Wind$	X
$x corr increm$	false	$Data$	$Prespos$	$Update$	04
$x drift$	false	$Data$	$Prespos$	$Update$	01
$y corr increm$	false	$Data$	$Prespos$	$Update$	05
$y drift$	false	$Data$	$Prespos$	$Update$	02
$z corr increm$	false	$Data$	$Prespos$	$Update$	06
$z drift$	false	$Data$	$Prespos$	$Update$	03

SS.PNL.CONFIG (continued)

A few panel configurations cannot be fully defined using the table above. Supplemental descriptions for a few configurations are listed below.

Value of !+pnl config+!	Definition
$alt baro AGL$	The panel may not be in this configuration if the system is in *A/A Guns* OR *A/A Manrip* OR *A/G Guns* OR *Manrip* OR *Walleye*.
$compfail$	The panel is in this configuration whenever @T(!+test_stage+! = PD) occurs. If @T(!+Comp fail+!) occurs before the PD test stage is exited, then the panel remains in the $compfail$ configuration until an intervening @T(!+init complete+!) occurs. If @T(!+Comp fail+!) does not occur before the PD test stage is exited, then the panel reverts to the configuration(s) that it was in before it entered $compfail$ configuration.
$dest lat$ $dest long$ $latitude$ $longitude$	These values are determined by modes and events, as well as by switch settings and keyboard inputs as described above. They appear in the table above, but the panel may also be in some of these configurations as described by Table SS.EVENT.12-b. In other words, the panel is a configuration in this list when the corresponding events/conditions in <u>either</u> table occur/become true.
$latitude error$ $longitude error$	The panel is in both of these configurations in any navigation update mode (as defined in [REQ], Section 3.0.1) <u>except</u> *AflyUpd*, when @T(!+TD pressed+!) occurs.
$none$	The panel is in this configuration when it is in none other.

SS.PNL.CONFIG (continued)

Term	Definition
$priority alt display$	The panel cannot be in this configuration if the system is in *A/A Guns* OR *A/A Manrip* OR *A/G Guns* OR *Manrip* OR *Walleye*

Table SS.PNL.CONFIG.12-b
Alternate definitions of $latitude$, $longitude$, $dest lat$, and $dest long$

MODES	EVENTS	
AflyUpd *HUDUpd* *RadarUpd* *FlyUpd* *TacUpd*	@T(In mode) OR @T(!+Enter pressed+!)	@T(!+dest selected+!)
MapUpd	@T(In mode) OR @T(!+Enter pressed+!)	X
Panel configurations:	$latitude$ $longitude$	$dest lat$ $dest long$

SS.PNL.FORMAT

The format rules are given in Section 4.6.0 of the Requirements. The "BLANK INTEGER" format implements the !label! format described in the Requirements; "REAL" implements !decimal point!. All other pairings are straightforward.

SS.PNL.INPUT

1. Values for the initial-value sysgen parameters are given in Section 4.6 of the Requirements.

2. This submodule uses the SS.PNL.CONFIG submodule to tell what item is currently being entered.

3. Any of the following will cause the error display:

   a. @T(!+Keybd pressed+!) twice, with no intervening occurrence of either @T(!+Enter pressed+!) or @T(!+Keybd input ready+!);

   b. @T(!+Enter pressed+!) when data is not legally enterable;

   c. @T(!+Keybd pressed+!) when the system is in no navigation update mode (as defined in [REQ], Section 3.0.1);

   d. !+data nbr pnl+! is not one of 00-26, 70-73, 80-85, or 88-99;

   e. @T(!+Keybd pressed+!) WHEN(!+in_flight+! and the current panel configuration is $latitude$ OR $longitude$);

   f. @T(!+Keybd pressed+!) WHEN((NOT !+in_flight+! OR !+adc tas up+!) and the current panel configuration is $wind dir$ OR $wind vel$);

   g. Pilot attempts to enter two-digit !+dest entry+! or three-digit !+data nbr pnl+!;

   h. Error in data format or range;

   i. @T(NOT !+TAC bearing valid+! OR NOT !+TAC range valid+! OR !+IMS mode+! = $Grid$ OR !+TAC range+! ls !+alt ADC+!) WHEN(!+in *TacUpd*);

   j. Number keyed in for !+dest entry+! is 0.

4. When the error display is shown, all format lights are turned off.

SS.SUBRTN

1. To calculate !+symbol_az_on_ASL+!, the current position and roation angle of the ASL must be known. The program calls the appropriate access programs directly.

SS.STAGE

This is a list of each stage of each alignment mode (as defined in [REQ], Section 3.0.1), identifying possible mode entries, mode exits, and repeated stages.

Alignment mode	Stages within the alignment mode						
Lautocal	CL	CA	fg	CA	ed	CA	nd
Sautocal	CL	CA	CA	ed	CA	nd	
OlUpdate	(fg	-TS-)					
HUDaln	$\overset{/}{\text{-HS-}}$	CL	CA	fg	(fg	-TS-)	
Landaln	CL	CA	fg	(fg	-TS-)		
SINSaln	-HS-	CL	CA	fg	-TS-		
Airaln	-FM-	CL	hl	$\underset{/}{\text{hg}}$	$\underset{/}{\text{(fg)}}$		

Legend (let "ST" denote an alignment stage):

Symbology	Meaning
$\overset{/}{\text{ST}}$	The mode may be exited when this stage is completed.
$\underset{/}{\text{ST}}$	The mode may be entered at the beginning of this stage.
( )	The enclosed stage, or sequence of stages, is repeated until the mode is exited.
ST	Large IMS axis adjustments are performed during this alignment stage.
st	Small IMS axis adjustments are performed during this alignment stage.
-ST-	No IMS axis adjustments are performed during this alignment stage.

SS.SYSVAL.DEVREAS

The following terms have the given implementation definitions:

!+adc alt up+!	true iff !+adc reasonable+! AND !+ADC alt valid+!.
!+adc reasonable+!	true iff 150 fps ls !+TAS ADC+! ls 1024 fps AND !+in flight+! AND !+ADC test result+!.
!+adc tas up+!	true iff !+adc reasonable+! AND !+ADC tas valid+!.
!+Doppler reasonable+!	!+DRS reliable+! for last 5 seconds AND 256 fps lseq !+gnd speed DRS+! lseq 1456 fps AND !+drift angle DRS+! lseq 29.5 degrees AND !+drift angle DRS+! change from .2 sec ago ls 4 deg AND !+gnd speed DRS+! change from .2 sec ago lseq 79 fps AND either !+drift angle DRS+! or !+gnd speed DRS+! has changed value in the last 2 seconds
!+Doppler up+!	!+Doppler reasonable+! AND !+DRS mode+! noteq $Memory$
!+IMS reasonable+!	!+speed total IMS+! lseq 1440 fps AND !+speed total IMS+! change from .2 sec ago lseq 50 fps.
!+mach reasonable+!	true iff !+mach ADC+! has not changed by more than .0195 in the last .2 seconds.
!+radalt reasonable+!	50 feet lseq !+alt Radar+! lseq 4990 feet AND ABSV(!+roll IMS+!) lseq 30 degrees
!+sr reasonable+!	1700 ft lseq !+Slt range FLR+! lseq 54000 ft AND ABSV(!+roll IMS+!) lseq 40 deg AND ABSV(!+FLR elev+!) ls 16 deg AND ABSV(!+FLR az+!) ls 16 deg AND the angle at which the FLR pointing line intersects the horizontal plane at the earth's surface is gt 4 degrees

SS.SYSVAL.IMSALN

The following terms have the given implementation definitions:

!+air velocity test passed+!	true iff difference between !+E vel IMS+! and !+e vel DRS+! and between !+N vel IMS+! and !+n vel DRS+! is lseq 5 fps. in *Airaln* mode.
!+drift test failed+!	true iff difference between old !+gyro drift delta n+! and the new !+gyro drift delta n+! gt 0.049.
!+drift test passed+!	true iff difference between old !+gyro drift delta n+! and new !+gyro drift delta n+! lseq 0.049.
!+elapsed navaln time+!	Initialized to zero whenever any of the following events occurs: @T(!+align_stage+! = $CL$) @T(!+Non-align+! = $Off$) when the system is in some alignment mode @T(!+align_stage+! = $FG$) WHEN(!+Non-align+! = $Off) @T(*OlUpdate*) @F(!+IMS mode+! = $Gndal$) When the time reaches #navaln_wraparound#, its value is reset to zero.
!+land velocity test failed+!	TO BE DETERMINED.
!+land velocity test passed+!	Becomes false upon entry into *Lautocal*, *Sautocal*, *Landaln*, *SINSaln*, *HUDaln*, *UDI*, *OLB*, *Mag sl*, *Grid*, *IMS fail*, or *Grtest* modes. Becomes true when !+E vel IMS+! and !+N vel IMS+! are both less than 1/8 fps when !+align_stage+! = $TS$.
!+nav velocity test failed+!	true iff difference between !+N vel IMS+! and !+n vel DRS+! and between !+E vel IMS+! and !+e vel DRS+! gt 10 fps.

!+SINS velocity test passed+!	Initial value: false. Becomes false upon entry into one of the following modes: *Lautocal*, *Sautocal*, *Landaln*, *SINSaln*, *HUDaln*, *UDI*, *OLB*, *Magsl*, *Grid*, *IMS fail*, *Grtest*. Becomes true when the difference between !+N vel IMS+! and !+SINS north vel+! and between !+E vel IMS+! and !+SINS east vel+! lseq 1 fps after 30 seconds in TS stage in *SINSaln* mode.

SS.SYSVAL.REFPT

The requirements-derived definitions for most of the items supplied by this module are given below. An item marked with "!!" brackets denotes an item that is not part of the interface (i.e., not provided by this submodule), but is defined somewhere else in the implementation notes.

!!adjusted point!!	The point overlayed by the HUD aiming symbol when @T(!+after slewing+!) occurs. Convert !+AS azimuth+! and !+AS elevation+! to position, bearing, and slant ranges by using Physical Model programs.
!+brg_ac_ftpt+! !+brg_ac_tgt+! !+brg_grtk_ap+!	Use the position (latitude, longitude) of the particular point in order to obtain the bearing by using Physical Models programs.
!+brg_grtk_cup+! !+brg_grtk_ftpt+! !+brg_grtk_oap+!	Use the position (latitude, longitude) of the particular point in order to obtain the bearing from the Physical Models module.
!!called-up point!!	The !!called-up point!! is defined when the system is in some navigation update mode (as defined in [REQ], Section 3.0.1) and @T(!+pnl config+! = $dest lat$ OR $dest long$) occurs. It is the location described by !+dest lat pnl+! and !+dest long pnl+!, parameterized by !+dest entry pnl+!.
!!corrected OAP!!	The point offset laterally from the OAP by bearing !+offset brg pnl+! and distance !+offset rng pnl+!; and offset vertically by distance !+offset dht pnl+!. The values !+offset brg pnl+!, !+offset rng pnl+!, and !+offset dht pnl+! are all parameterized by !+Fly to num+!.

!+desig+! Defined by the table below.

MODES	EVENTS	
HUDaln All navigation update modes	@T(!+TD pressed+!)	@T(In mode) OR @F(In mode)
BOC *SBOC*	@T(In mode) WHEN(NOT !!SK!!)	X
BOCFlyto0 (when NOT !!SK!!)	@T(!+TD pressed+! OR !+RE pressed+! OR !+after slewing+!) WHEN(NOT !+desig+!)	@T(In mode) OR @T(!+TD pressed+!) WHEN(!+desig+!)
BOCoffset	@T(!+TD pressed+! OR !+RE pressed+!) WHEN(NOT !+desig+!)	@T(In mode) OR @T(!+TD pressed+!) WHEN(!+desig+!)
Nattack (when NOT !!SK!!)	@T(!+TD pressed+! OR !+RE pressed+! OR !+after slewing+!) WHEN(NOT !+desig+!)	@F(In mode) OR @T(!+TD pressed+!) WHEN(!+desig+!) OR @T(In mode) WHEN (NOT !!just left a BOC!!)
Noffset (when NOT !!SK!!)	@T(!+TD pressed+! OR !+RE pressed+!) WHEN(NOT !+desig+!)	@F(In mode) OR @T(!+TD pressed+!) WHEN(!+desig+!) OR @T(In mode) WHEN (NOT !!just left a BOC!!)
NBShrike	@T(!+TD pressed+! OR !+RE pressed+!) WHEN(NOT !+desig+!)	@F(In mode) OR @T(!+TD pressed+!) WHEN(!+desig+!)
SBOCFlyto0	@T(!+TD pressed+! OR !+after slewing+!) WHEN(NOT !+desig+!)	@T(In mode)
SBOCoffset	@T(!+TD pressed+!) WHEN(NOT !+desig+!)	@T(In mode)
Snattack	@T(!+TD pressed+! OR !+after slewing+!) WHEN(NOT !+desig+!)	@F(In mode)
Snoffset	@T(!+TD pressed+!) WHEN(NOT !+desig+!)	@T(In mode) OR @F(In mode)
Walleye	@T(!+TD pressed+!) WHEN(NOT !+desig+!)	@F(In mode) OR @T(!+TD pressed+!) WHEN(!+desig+!)
Value of !+desig+!:	true	false

!+dest lat+!	1) When @T(!+WAYPT available+!) occurs, use the value !+WAYPT lat+! to update the destination latitude parameterized by !+WAYPT ID+!; 2) As soon as a new value of !+dest lat pnl+! is entered via the panel, use it to update the destination latitude parameterized by !+dest entry pnl+!; 3) When the following sequence of events occurs, use the value !+Map latitude+! to update the destination latitude parameterized by !+dest entry pnl+!: a) @T(!+pnl config+! = $dest lat$) b) @T(!+Map hold+!) c) @T(!+pnl input complete+!)
!+dest long+!	1) When @T(!+WAYPT available+!) occurs, use the value !+WAYPT long+! to update the destination longitude parameterized by !+WAYPT ID+!; 2) As soon as a new value of !+dest long pnl+! is entered via the panel, used it to update the destination longitude parameterized by !+dest entry pnl+!; 3) When the following sequence of events occurs, use the value !+Map longitude+! to update the destination longitude parameterized by !+dest entry pnl+!: a) @T(!+pnl config+! = $dest long$) b) @T(!+Map hold+!) d) @T(!+pnl input complete+!)
!!fix point AS!!	The point on the ground overlayed by the HUD aiming symbol at the time @T(!+desig+!) OR @T(!+after slewing+!) occurs.
!!fix point AS desig!!	The point on the ground overlayed by the HUD aiming symbol at the time @T(!+desig+!) occurs.

!!fix point FLR!!	The point on the ground overlayed by the FLR cursors at the time @T(!+after slewing+!) occurs.
!!fix point PMDS!!	The point displayed by the map at the time @T(!+TD pressed+!) occurs.
!!fly-to point!!	If !+Fly to state+! = $Mark$ then the !!fly-to point!! is defined as the position described by !+mark lat+! and !+mark long+!, parameterized by !+Fly to num+!. If !+Fly to state+! = $Dest$ then the !!fly-to point!! is defined as the position described by !+dest lat+! and !+dest long+!, parameterized by !+Fly to num+!.
!+gr_ac_ftpt+! !+gr_ac_fxpt+! !+gr_ac_HUDrefpt+! !+gr_ac_oap+! !+gr_ac_tgt+!	Use the position (latitude, longitude) of the particular point to obtain the ground range from the Physical Models module.
!+ground danger+!	Becomes <u>true</u> when !+sr_ac_gpup+! becomes 0.
!+HUDrefpt_az+! !+HUDrefpt_elev+!	Use the position (latitude, longitude) of the !!HUD reference point!! to obtain the angles. If the current !!HUD reference point!! is a point on the ground track 8 nmi ahead of the a/c, then the azimuth is equal to !+drift angle+!.

!!HUD reference point!! Defined by the following tables.

Definition of !!HUD reference point!! in certain weapon modes

MODES	CONDITIONS				
HUDdown1 *Nattack* *SHUDdown1* *Snattack*	X	X	X	Always	X
HUDdown2 *Noffset* *SHUDdown2* *Snoffset*	X	X	NOT !+desig+!	!+desig+!	X
BOC *SBOC*	X	X	X	Always	
BOCFlyto0 *SBOCFlyto0*	X	X	X	!+desig+!	NOT !+desig+!
BOCoffset *SBOCoffset*	X	NOT !+desig+! AND !+before slewing+!	NOT !+desig+! AND !+after slewing+!	!+desig+!	X
A/A Guns *A/A Manrip* *A/G Guns*	Always	X	X	X	X
!!HUD reference point!!:	!!impact point!!	!!fly-to-point!!	!!offset aim point!!	!!target!!	8 nmi ahead of a/c on !+grtk+!

Definition of !!HUD reference point!! in non-weapon modes

MODES	CONDITIONS		
HUDaln	X	X	Always
HUDUpd *RadarUpd*	NOT !+after slewing+!	!+after slewing+!	X
!!HUD reference point!!:	!!called-up point!!	!!fix point AS!!	!!adjusted point!!

!!impact point!!	Point defined by !+ip lat+! and !+ip long+!.
!!just left a BOC!!	less than 1.6 seconds have elapsed since the most recent exit from *BOC*, *BOCoffset*, or *BOCFlyto0* mode.
!+latitude+!	Defined by the table below. This table tells when to change the source of the latitude value.

MODES	EVENTS	VALUE OF !+latitude+!
Sautocal *SINSaln*	@T(In mode)	!+SINS lat+!
	@T(!+New latitude pnl entered+!)	!+latitude pnl+!
HUDaln *Landaln* *Lautocal* *OlUpdate*	@T(!+New latitude pnl entered+!)	!+latitude pnl+!
Airaln *DI* *DIG* *PolarDI* *UDI*	@T(In mode)	current latitude computed by Physical Models module
Grid *I* *Mag sl* *OLB*	@T(!+New latitude pnl entered+!) WHEN(NOT !+in flight+!)	!+latitude pnl+!
	@T(In mode) OR @T(!+in flight+!)	latitude computed by Physical Models module

!+latitude_cup+!	This is the latitude of the !!called-up point!!.
!+latitude error+!	This is the difference between the first and second reference latitudes (measured from the first to the second), as defined in Table SS.SYSVAL.REFPT.12-a.
!+longitude+!	Defined by the table below. This table tells when to change the source of the longitude value.

Definition of !+longitude+!

MODES	EVENTS	VALUE OF !+longitude+!
Sautocal *SINSaln*	@T(In mode)	!+SINS long+!
	@T(!+new longitude pnl entered+!)	!+longitude pnl+!
HUDaln *Landaln* *Lautocal* *OlUpdate*	@T(!+new longitude pnl entered+!)	!+longitude pnl+!
Airaln *DI* *DIG* *PolarDI* *UDI*	@T(In mode)	Current longitude computed by Physical Models module
Grid *I* *Mag sl* *OLB*	@T(!+new longitude pnl entered+!) WHEN(NOT !+in flight+!) AND	!+longitude pnl+!
	@T(In mode) OR @T(!+in flight+!)	Current longitude computed by Physical Models module

!+longitude_cup+!	This is the longitude of the !!called-up point!!.
!+longitude error+!	This is the difference between the first and second reference longitudes (measured from the first to the second), as defined in Table SS.SYSVAL.REFPT.12-a.

Table SS.SYSVAL.REFPT.12-a
Reference points of !+latitude error+! and !+longitude error+!

MODES	First reference point	Second reference point
HUDUpd	!!called-up point!!	!!fix point AS!!
RadarUpd	!!called-up point!!	!!fix point FLR!!
FlyUpd	!!called-up point!!	!+latitude+! & !+longitude+!
TacUpd	!+latitude+! & !+longitude+!	Position of the a/c computed by assuming that the !!called-up point!! is at !+TAC range+! and !+TAC bearing+! from the a/c. Use PM position offset program.
MapUpd	!+latitude+! & !+longitude+!	!!fix point PMDS!!

!+mark+!	Initialized to zero at system-generation time. Thereafter, each time @T(!+Mark pressed+!) occurs, !+Mark+! is incremented thus:

```
                if !+Mark+! = 9
                   then !+Mark+!  :=  1
                   else !+Mark+!  :=  !+Mark+!  +  1
```

!+mark lat+!	!+latitude+! at the time @T(!+Mark pressed+!) occurs. This value will be associated with the value of !+Mark+! after @T(!+Mark pressed+!) occurs.
!+mark long+!	!+longitude+! at the time @T(!+Mark pressed+!) occurs. This value will be associated with the value of !+Mark+! after @T(!+Mark pressed+!) occurs.
!!offset aim point!! (OAP)	Defined by the table below.

Definition of !!offset aim point!! (OAP)

MODES	Offset Aim Point
BOCoffset *SBOCoffset*	!!fly-to point!! if NOT !+after slewing+!; !!fix point FLR!! otherwise
Noffset *Snoffset*	!!fix point AS!!
HUDdown2 *SHUDdown2*	!!point ahead desig!!
Any other weapon mode, or no weapon mode	!!target!!

!!point ahead desig!!	The point on the ground intersecting the aircraft's Ya axis at the time @T(!+desig+!) occurs.

!!SK!!	!+Weapon Class+! = SK
!+sr_ac_ap+! !+sr_ac_cup+! !+sr_ac_ftpt+! !+sr_ac_oap+! !+sr_ac_tgt+	Use the position (latitude, longitude) of the particular point and use Physical Models programs to obtain the slant range.
!+sr_ac_gpup+!	Use Physical Models programs to determine the position of the ground pullup point, and then the slant range from the aircraft to it.
!+steering error to rls+!	Use Physical Models programs to determine the release point and provide a bearing to that location.
!+steering error to tgt+!	!+brg_grtk_tgt+!, if lseq 180 deg; !+brg_grtk_tgt+! - 360, if gt 180 deg.
!+time to ftpt+!	Use Physical Models programs to obtain the time, given the position of the !!fly-to point!! and the current magnitudes of !+velocity east system+! and !+velocity north system+!.

!!target!! Defined by the table below.

MODES	Definition of !!target!!
BOCFlyto0 *Nattack* *SBOCFlyto0* *Snattack*	!!fix point AS!!
Walleye	!!fix point AS desig!!
BOCoffset *HUDdown2* *Noffset* *SBOCoffset* *SHUDdown2* *Snoffset*	!!corrected OAP!!
BOC *SBOC*	!!fly-to point!! if NOT !+after slewing+!; !!fix point FLR!! otherwise
HUDdown1 *SHUDdown1*	!!point ahead desig!!
A/A Guns *A/A Manrip* *A/G Guns* *CCIP*	!!impact point!!
No weapon mode	!!fly-to point!!

SS.SYSVAL.SLEW

Displacements should be computed using the following rates:

Radar rate:	
Lateral (in degrees per second)	25/64 x !+Slew right-left+! x ((4 x !+Slew right-left+!^{2}) + 36)
Vertical (in feet per second)	25 x 16 x !+Slew up-down+! x ((4 x !+Slew up-down+!^{2}) + 36)
HUD rate:	
Lateral (in degrees per second)	25 x !+Slew right-left+! x ((!+Slew right-left+!^{2})/256) + .035
Vertical (in degrees per second)	25 x !+Slew up-down+! x (((!+Slew up-down+!^{2}) / 256) + .035)
Map rate:	
Latitude displacement (earth arc-secs per sec)	25 x !+Slew *right-left+*! x (((!+Slew right-left+!^{2}) / 4) + 2.25)
Longitude displacement (earth arc-secs per sec)	25 x !+Slew up-down+! x (((!+Slew up-down+!^{2}) / 4) + 2.25)

The slew-rate equations include a factor of 25 that converts a delta-slew (for one compute cycle) into a slew-movement-per-second rate. This assumes 25 cycles per second, and the rates must be adjusted (parameterized) to reflect what the periodicity of our program will actually be.

!+HUD slew legal+! is defined by following table.

MODES	CONDITIONS	
Nattack	NOT !!rls imminent!! AND NOT !!SK!! OR !!desig retent!! AND !!SK!!	!!rls imminent!! AND NOT !!SK!! OR NOT !!desig retent!! AND !!SK!!
Snattack *Noffset* *Snoffset*	NOT !!rls imminent!!	!!rls imminent!!
HUDUpd *RadarUpd*	Always	X
A/A Guns *A/A Manrip* *A/G Guns* *HUDdown1* *HUDdown2* *SHUDdown1* *SHUDdown2* *Walleye*	X	Always
BOC *BOCFlyto0* *BOCoffset* *SBOC* *SBOCFlyto0* *SBOCoffset*	NOT !!rls imminent!! AND !+gr_ac_HUDrefpt+! lseq 20 nmi	!!rls imminent!! OR !+gr_ac_HUDrefpt+! gt 20 nmi
Value:	<u>true</u>	<u>false</u>

!!desig retent!!	true iff !+desig+! is true, and became true when the system was in *BOC*, *BOCoffset*, or *BOCFlyto0* modes.
!!rls imminent!!	In modes *Nattack*, *Noffset*, *BOC*, *BOCoffset*, or *BOCFlyto0*: true iff either solution cue is between the top and center of the HUD flight path marker; i.e., a solution cue elevation is above the FPM, but by not more than .59 degrees. In modes *Snattack*, *Snoffset*, *SBOC*, *SBOCoffset*, or *SBOCFlyto0*: true iff the lower solution is above the flight path marker, but by not more than .59 degrees.
!!SK!!	!+Weapon Class+! = SK

SS.SYSVAL.VALSEL

!+alt priority stale+!
!+alt priority source+! When and how these are updated is defined by the table below.

Definition of !+alt priority stale+!
and !+alt priority source+!

MODES		CONDITIONS
Nattack *Noffset* *Snattack* *Snoffset* *HUDdown1* *HUDdown2* *SHUDdown1* *SHUDdown2*	@T(In mode) OR @F(!+desig+!)	@T(!+desig+!) OR @F(!+Slew displacement non-zero+!) WHEN(!+rls_pts_passed+! = 0) OR @T(!+sr reasonable+!) WHEN(!+rls_pts_passed+! = 0 AND !+desig+!)
BOCFlytoO *SBOCFlytoO*	@T(In mode) OR @F(!+desig+!)	@T(!+desig+!) OR @F(!+Slew displacement non-zero+!) WHEN(!+rls_pts_passed+! = 0)
BOC *SBOC*	@T(In mode AND !+gr_ac_tgt+! gt 30 nmi)	@T(!+gr_ac_tgt+! lseq 30 nmi) OR @F(!+Slew displacement non-zero+!) WHEN (!+gr_ac_tgt+! lseq 20 nmi AND !+rls_pts_passed+! = 0)
BOCoffset *SBOCoffset*	@T(In mode) OR @F(!+desig+!) OR @T(!+gr_ac_oap+! gt 30 nmi)	@T(!+desig+!) OR @F(!+Slew displacement non-zero+!) WHEN(!+gr_ac_oap+! lseq 20 nmi AND !+rls_pts_passed+! = 0) OR @T(!+gr_ac_oap+! lseq 30 nmi)
CCIP	@T(In mode)	@T(!+ip_elev+! lseq 16 deg) WHEN(!+rls_pts_passed+! = 0)
Not in any weapon mode listed above	@T(In mode)	X
!+alt priority stale+!:	0	!+alt priority ranging+!
!+alt priority source+!:	$None$	Depends on sensor source of !+alt priority ranging+!; H if ADC, $F$ if from FLR slant range, and $A$ if radar altimeter.

!+alt priority ranging+! Defined by the table below.

Definition of !+alt priority ranging+!

MODES	CONDITIONS		
CCIP *HUDdown1* *HUDdown2* *Nattack* *Noffset* *SHUDdown1* *SHUDdown2* *Snattack* *Snoffset*	!+sr reason-able+!	NOT !+sr reasonable+! AND !+radalt priority pnl+!	NOT !+sr reasonable+! AND NOT !+radalt priority pnl+!
BOC *BOCFlyto0* *BOCoffset* *SBOC* *SBOCFlyto0* *SBOCoffset*	X	!+radalt priority pnl+!	NOT !+radalt priority pnl+!
!+Alt priority ranging+!:	!+alt from sr+!	!+alt RADAR+!	!+alt ADC+!

!+drift angle+! If !+Doppler up+! then !+drift angle DRS+!; otherwise !+drift angle IMS+!.

!+in_flight+! Use value of !+WOG+!, from the DIM.

!+velocity east system+!
!+velocity north system+! Defined by the tables below.

Definition of !+velocity east system+!
and !+velocity north system+! in Navigation modes

MODES	CONDITIONS			
DI *DIG* *PolarDI* *UDI*	Always	X	X	X
I *PolarI*	X	Always	X	X
Grid *IMS fail* *Magsl* *OLB*	X	X	!+Doppler up+!	NOT !+Doppler up+!
!+velocity north system+!: Based on:	!+N vel IMS+! damped by !+gnd speed DRS+!	!+N vel IMS+!	!+gnd speed DRS+! system heading	!+TAS ADC+! !+AOA+! !+pitch IMS+! !+wind dir+! !+wind vel+!
!+velocity east system+!: Based on:	!+E vel IMS+! damped by !+gnd speed DRS+!	!+E vel IMS+!	!+gnd speed DRS+! system heading	!+TAS ADC+! !+AOA+! !+pitch IMS+! !+wind dir+! !+wind vel+!

Definition of !+velocity east system+!
and !+velocity north system+! in Alignment modes

MODES	!+velocity north system+!	!+velocity east system+!
HUDaln *Lautocal* *Landaln* *OlUpdate*	0	0
Sautocal *SINSaln*	!+SINS north vel+! adjusted by SINS x, y, and z offsets	!+SINS east vel+! adjusted by SINS x, y, and z offsets
Airaln	!+N vel IMS+! damped by !+gnd speed DRS+!	!+E vel IMS+! damped by !+gnd speed DRS+!

¦+velocity vertical system+¦ Defined by the table below.

Definition of ¦+velocity vertical system+¦

MODES	CONDITIONS			
Airaln				
DI *DIG* *I* *PolarDI* *PolarI* *UDI*	¦+ADC alt valid+¦	NOT ¦+ADC alt valid+¦	X	X
Grid *Magsl* *OLB*	X	X	¦+adc tas up+¦	NOT ¦+adc tas up+¦
IMS fail	X	X	X	Always
Any alignment mode except *Airaln*	¦+ADC alt valid+¦	NOT ¦+ADC alt valid+¦	X	X
¦+velocity vertical system+¦: based on:	¦+V vel IMS+¦ damped by ¦+alt ADC+¦	sys. hor. vels. flt path angle	¦+TAS ADC+¦ flt path angle	0

Programmers should invoke the appropriate Physical Models programs to return vertical velocity based on the specified models.

!+wind dir+! The source of the measurement alternates between the most recent panel entry and the value computed by the Physical Model. When the source changes is defined by the table below.

Definition of !+wind dir+!

MODES	EVENTS	
All alignment modes except *Airaln*	@T(In mode)	X
Airaln *DI* *DIG* *Grid* *I* *IMS fail* *Mag sl* *OLB* *PolarDI* *PolarI* *UDI*	@T(In mode) OR @T(!+adc tas up+! OR NOT !+in flight+!)	@T(!+new wind dir pnl entered+!) WHEN(!+in flight+! AND NOT !+tas adc up+!)
!+wind dir+!:	wind direction computed by Physical Models module	!+wind dir pnl+!

!+wind vel+! The source of the measurement alternates between the most recent panel entry and the value computed by the Physical Model. When the source changes is defined by the table below.

Definition of !+wind vel+!

MODES	EVENTS	
All alignment modes except *Airaln*	@T(In mode)	X
Airaln *DI* *DIG* *Grid* *I* *IMS fail* *Mag sl* *OLB* *PolarDI* *PolarI* *UDI*	@T(In mode) OR @T(!+adc tas up+! OR NOT !+in flight+!)	@T(!+new wind vel pnl entered+!) WHEN(!+in flight+! AND NOT !+tas adc up+!)
!+wind vel+!:	wind speed computed by Physical Models module	!+wind vel pnl+!

SS.SYSVAL.WPNRLS

!+blast danger+!	Becomes true when !+sr_ac_btpup+! becomes 0.
!+computed rls+!	Signalled according to the table below. The terms bracketed by "!!" marks are explained under their own implementation notes elsewhere in this section.

Definition of !+computed rls+!

MODES	EVENTS
NBnotShrike	@T(!+Special solution+!) WHEN(!+rls_pts_passed+! = 0) OR @T(!!stik distance achieved!!) WHEN(!+rls_pts_passed+! gteq 1 AND !+rls_pts_passed+! ls !+stik_quan+!)
NBShrike	@T(!!weapon prepared!! AND !+SK solution+!)
HiNuke	@T(!+special solution+!)
LoNuke	@T(!+flt path angle+! = !+Amax+!) WHEN (!!eligible release!! AND NOT !!special soln occurred!!) OR @T(!+flt path angle+! = !+Amin+!) WHEN(!!eligible release!! AND !!special soln occurred!!) OR @T(!+special solution+!) WHEN(!!eligible release!! AND !+Amin+! lseq !+flt path angle+! lseq !+Amax+!) OR @T(!!eligible release!! AND time since @T(!+special solution+!) occurred lseq 2 seconds)
A/A Manrip *CCIP* *Manrip* *Walleye*	@T(!!stik distance achieved!!) WHEN(!+rls_pts_passed+! gteq 1 AND !+rls_pts_passed+! ls !+stik_quan+!)

!!eligible release!!	!+desig+! AND !+PUAC mode+! = $On$ OR $Intermittent$

!+GAS+!
!+OTS+!
!+steering to tgt+!
!+TOS+! All are defined by the following tables.

Definition of !+GAS+!

MODES	EVENTS	
LoNuke	@T(!+tgt ahead+!) WHEN(!+gr_ac_lastip+! lseq 42 nmi)	@T(!+OTS+!) OR @T(!+TOS+!) OR @T(!+steering to tgt+!) OR @F(In mode)
HiNuke	@T(!+gr_ac_lastip+! gt 13000 ft) WHEN(!+desig+! AND !+wpns_rlsd+! = 0 AND !+rls_pts_passed+!=!+stik quan+!) OR @T(!+tgt ahead+!) WHEN(!+gr_ac_lastip+! lseq 42nmi)	@T(!+OTS+!) OR @T(!+TOS+!) OR @T(!+steering to tgt+!) OR @F(In mode)
NBnotShrike	@T(.+gr_ac_lastip+! gt 13000 ft) WHEN(!+desig+! AND !+rls_pts_passed+!=!+stik quan+!)	@T(!+OTS+!) OR @T(!+TOS+!) OR @T(!+steering to tgt+!) OR @F(In mode)
NBShrike *Walleye*	@T(!+stik empty+!) WHEN(!+desig+!)	@T(!+OTS+!) OR @T(!+TOS+!) OR @T(!+steering to tgt+!) OR @F(In mode)
Value of !+GAS+!:	<u>true</u>	<u>false</u>

AD-A118 897 NAVAL RESEARCH LAB WASHINGTON DC F/G 9/2
INTERFACE SPECIFICATIONS FOR THE A-7E SHARED SERVICES MODULE.(U)
SEP 82 P C CLEMENTS
UNCLASSIFIED NRL-MR-4863 NL

2 of 2

END
DATE
FILMED
10-82
DTIC

Definition of !+OTS+!

MODES	EVENTS	
LoNuke	@T(!+stik empty+!) WHEN(!+desig+! AND !+wpns_rlsd+! = 0)	@T(!+TOS+!) OR @T(!+GAS+!) OR @T(!+steering to tgt+!) OR @F(In mode)
NBnotShrike	@T(!+stik empty+!) WHEN(!+desig+!)	@T(!+TOS+!) OR @T(!+GAS+!) OR @T(!+steering to tgt+!) OR @F(In mode)
Value of !+OTS+!:	true	false

Definition of !+steering to tgt+!

MODES	EVENTS	
LoNuke **HiNuke** **NBnotShrike** **NBShrike** *Walleye*	@T(!+desig+! OR @T(!+gr_ac_tgt+! lseq 5000 ft OR ABS(!+steering error to tgt+! lseq 20 degrees) WHEN(!+GAS+!)	@T(!+OTS+!) OR @T(!+TOS+!) OR @T(!+GAS+!) OR @F(In mode)
Value of !+steering to tgt+!:	true	false

Definition of !+TOS+!

MODES	EVENTS	
LoNuke	@T(!+wpns_rlsd+! = 1)	@T(!+OTS+!) OR @T(!+GAS+!) OR @T(!+steering to tgt+!)
HiNuke	@T(!+stik empty+!) WHEN(!+desig+!) OR @T(!+wpns_rlsd+! = 1)	@T(!+OTS+!) OR @T(!+GAS+!) OR @T(!+steering to tgt+!)
!+TOS+!:	true	false

!+high drag release+!	!+Weapon Class+! = (SH OR HD) OR (!+Weapon Class+! = (OD OR OR OR SOD) AND !+High Drag+!)
!+low drag release+!	!+Weapon Class+! = (SL OR SSL) OR (!+Weapon Class+! = (OD OR OR OR SOD) AND NOT !+High Drag+!)
!!MRI dist!!	The distance along the ground between two successive impact points, assuming a time interval between successive releases given by the MRI curves in Section 2.4 of the Requirements. To compute an MRI for a given weapon, the !+Weapon Class+! and current !+norm accel+! must be known.
!+rls_pts_passed+!	Initialized to 0 whenever one of the following occurs: @T(!+desig+!) @T(*CCIP* OR *Manrip* OR *A/A Manrip*) @T(!+RE pressed+!) WHEN(!+rls_pts_passed+! = !+stik_quan+!) Incremented by 1 whenever @T(!+computed rls+!) occurs.
!+r65+!	!+desig+! AND !+low drag release+! AND !+Weapon Class+! = SOD OR SSH AND !+tgt ahead+! AND a/c at optimum pullup range for a 65 degree flight path angle at release.
!+rmax+!	!+desig+! AND !+low drag release+! AND !+Weapon Class+! = SOD OR SSH AND !+tgt ahead+! AND a/c at maximum pullup range that would result in weapon impact on target.
!+rmax+6000+!	!+desig+! AND !+low drag release+! AND !+Weapon Class+! = SOD OR SSH AND !+tgt ahead+! AND a/c at pullup range that would result in weapon impact 6000 feet short of target.

!+rmin+!	!+desig+! AND !+low drag release+! AND !+Weapon Class+! = SOD OR SSH AND !+tgt ahead+! AND a/c at minimum pullup range that would result in weapon impact on the target.
!+rmin+6000+!	!+desig+! AND !+low drag release+! AND !+Weapon Class+! = SOD OR SSH AND !+tgt ahead+! AND a/c at pullup range that would result in weapon impact 6000 feet long of target.
!+special in range+!	if !+low drag release+! then true iff !+gr_ac_tgt+! lseq 10 nmi; if !+high drag release+! then true iff !+target in range+!.
!+special solution+!	!+miss distance+! lseq 10 feet AND ABS(!+steering error to tgt+!) lseq 20 deg. A likely change would be to make these criteria contingent upon the chosen weapon type.
!!special soln occurred!!	true iff the event @T(!+Special solution+!) has occurred since the last target designation.
!+sr_ac_btpup+!	Use Physical Models programs to return the position of the blast pullup point and then the slant range. The radius of avoidance is 1500 feet; note that a likely change would be to make the radius weapon-dependent.
!+sr_ac_ip+!	Use Physical Models programs to obtain the position of, and then the slant range to, the impact point.
!+sr_ac_rls+!	Use Physical Models programs to obtain the position of, and then the slant range to, the release point.

!!stik distance achieved!!

true when the ground range from the last (previous) computed impact point to the next computed impact point is equal to the required distance. The distance is a function of !+Weapon Class+!, !!MRI dist!!, and !+Weap Interval+!. The distance is not computed for weapon classes GN, RK, SSH, SOD, or WL, since these weapons are not delivered in stiks.

!+Weapon Class+!	Distance
SK	1 nmi
MF	MAX(!!MRI dist!!, !+Weap Interval x 10)
Others	MAX(!!MRI dist!!, !+Weap Interval+!)

A change in the value of !+Weap Interval+! should be ignored if it occurs during a stik delivery when !+RE pressed+!.

!+stik_quan+!

```
if !+Weapon Class+! = $SOD$ OR $SSH$ OR $WL$ then 1
if !+Weapon Class+! = $SK$ then
          MIN( !+Weap Quantity+!, 4 );
if !+Weapon Class+! = $MF$ then
          MIN( !+Weap Quantity+!, 20);
if !+Weapon Class+! = none of the above, then
          MAX( !+Weap Quantity+!, 1 ).
```

A change in the value of !+Weap Quantity+! should be ignored if it occurs during a stik delivery when !+RE pressed+!.

!+target in range+!

See definition of !target in range! in the Requirements document. See also the Device Interface Module Weapon Characteristics interface.

!+time to prepare+!

!+time to rls+! lseq !+preparation time+!.

!!weapon prepared!!

elapsed time since last call to +DI_PREPARE_WEAPON+ gteq !+preparation time+! for the current weapon.

!+wpns rlsd+!

Initialized to zero whenever the event @T(!+stik created+!) occurs. Incremented by one whenever @T(!+Rel in Progress+!) occurs.

APPENDIX 3

Shared Services Module Interface Term Index

with Mapping to Requirements

The following is a list of all values (either conditions or events) produced by the Shared Services module. Following each is the name of the submodule the produces it, and a reference to that part of the Requirements document that corresponds to the term.

Entry name	Producing Submodule	Mapping to Requirements
!+ACAIRB+!	DIAGIO	/ACAIRB/
!+AC stage complete+!	STAGE	!AC1 Tstage!, !AC2 Tstage!
!+ADCFAIL+!	DIAGIO	/ADCFAIL/
!+adc alt up+!	SYSVAL.DEVREAS	!ADC up!
!+adc reasonable+!	SYSVAL.DEVREAS	!ADC Reasonable!
!+adc tas up+!	SYSVAL.DEVREAS	!ADC up!
!+after slewing+!	SYSVAL.SLEW	!After slewing!
!+air velocity test passed+!	SYSVAL.IMSALN	!Air velocity test passed!
!+align_mode+!	MODE	Section 3.2
!+align_stage+!	STAGE	Section 3.2.0
!+alt priority ranging+!	SYSVAL.VALSEL	Section 4.6.38
!+alt priority source+!	SYSVAL.VALSEL	Section 4.6.38
!+alt priority stale+!	SYSVAL.VALSEL	Section 4.6.38
!+ap ahead+!	SYSVAL.ACPOSNREL	!adjusted point!
!+ARPINT+!	DIAGIO	/ARPINT/
!+ARPPAIRS+!	DIAGIO	/ARPPAIRS/
!+ARPQUANT+!	DIAGIO	/ARPQUANT/
!+az miss dist+!	SYSVAL.WPNRLS	!azimuth miss distance!
!+az ref hdg pnl+!	PNL.INPUT	!azimuth reference heading!; see also Sections 4.6.2, 4.6.10
!+before slewing+!	SYSVAL.SLEW	!Before slewing!
!+blast danger+!	SYSVAL.WPNRLS	!blast pullup point!
!+BMBDRAG eq High+!	DIAGIO	/BMBDRAG/
!+brg_ac_ftpt+!	SYSVAL.ACPOSNREL	!Fly-to point!
!+brg_ac_tgt+!	SYSVAL.ACPOSNREL	!target!
!+brg_grtk_ap+!	SYSVAL.ACPOSNREL	!adjusted point!
!+brg_grtk_cup+!	SYSVAL.ACPOSNREL	!Called-up point!

Entry name	Producing Submodule	Mapping to Requirements
!+brg_grtk_ftpt+!	SYSVAL.ACPOSNREL	!Fly-to point!
!+brg_grtk_oap+!	SYSVAL.ACPOSNREL	!OAP!, !target!
!+brg_grtk_tgt+!	SYSVAL.ACPOSNREL	!target!
!+burst ht pnl+!	PNL.INPUT	Section 4.6.14
!+CA stage complete+!	STAGE	!CA stage!
!+CA2 stage complete+!	STAGE	end of !CA stage!
!+central long a pnl+!	PNL.INPUT	Sections 4.6.2, 4.6.27
!+central long b pnl+!	PNL.INPUT	Sections 4.6.2, 4.6.27
!+CL stage complete+!	STAGE	!CL stage!
!+CL2 stage complete+!	STAGE	end of !CL stage!
!+computed rls+!	SYSVAL.WPNRLS	Section 4.4
!+CS stage complete+!	STAGE	!CS Tstage!
!+cup ahead+!	SYSVAL.ACPOSNREL	!Called-up point!
!+data enterable+!	PNL.INPUT	Section 4.6.2
!+data nbr pnl+!	PNL.INPUT	Sections 4.6.18-4.6.54
!+DC stage complete+!	STAGE	!DC Tstage!
!+desig+!	SYSVAL.FIXLOCN	!desig!
!+dest altitude pnl+!	PNL.INPUT	!Destaltitude!; see also Sections 4.6.2, 4.6.12
!+dest entry pnl+!	PNL.INPUT	Section 4.6.2; see also !dest called up!
!+dest lat+!	SYSVAL.FIXLOCN	Section 4.6.11
!+dest lat pnl+!	PNL.INPUT	Sections 4.6.2, 4.6.11
!+dest long+!	SYSVAL.FIXLOCN	Section 4.6.11
!+dest long pnl+!	PNL.INPUT	Sections 4.6.2, 4.6.11
!+dest mslp pnl+!	PNL.INPUT	Sections 4.6.2, 4.6.12
!+dest selected+!	PNL.INPUT	!dest called up!
!+DIO stage complete+!	STAGE	!DIO Tstage!
!+Doppler coupled pnl+!	PNL.INPUT	!Doppler coupled!; see also Sections 4.6.2, 4.6.32
!+Doppler reasonable+!	SYSVAL.DEVREAS	!Doppler Reasonable!
!+Doppler up+!	SYSVAL.DEVREAS	!Doppler Up!
!+drift angle+!	SYSVAL.VALSEL	!drift angle!
!+drift test failed+!	SYSVAL.IMSALN	!New 01 test passed!
!+drift test passed+!	SYSVAL.IMSALN	!New 01 test passed!
!+DRSREL+!	DIAGIO	/DRSREL/
!+during slewing+!	SYSVAL.SLEW	!During slewing!
!+e coarse bias pnl+!	PNL.INPUT	Sections 4.6.2, 4.6.25
!+e coarse scale pnl+!	PNL.INPUT	Sections 4.6.2, 4.6.23
!+e fine bias pnl+!	PNL.INPUT	Sections 4.6.2, 4.6.26
!+e fine scale pnl+!	PNL.INPUT	Sections 4.6.2, 4.6.24
!+ED stage complete+!	STAGE	!ED stage!
!+ED2 stage complete+!	STAGE	end of !ED stage!

Entry name	Producing Submodule	Mapping to Requirements
!+elapsed navaln time+!	SYSVAL.IMSALN	Section 4.6.44
!+FG stage complete+!	STAGE	!FG stage!
!+FM stage complete+!	STAGE	!FM stage!
!+ftpt ahead+!	SYSVAL.ACPOSNREL	!Fly-to point!
!+GAS+!	SYSVAL.WPNRLS	!GAS!
!+GA stage complete+!	STAGE	!GA Tstage!
!+gr_ac_ftpt+!	SYSVAL.ACPOSNREL	!Fly-to point!
!+gr_ac_fxpt+!	SYSVAL.ACPOSNREL	!fixpoint!
!+gr_ac_HUDrefpt+!	SYSVAL.ACPOSNREL	Section 4.3.1
!+gr_ac_oap+!	SYSVAL.ACPOSNREL	!OAP!, !target!
!+gr_ac_rmax+!	SYSVAL.WPNRLS	!Rmax!
!+gr_ac_tgt+!	SYSVAL.ACPOSNREL	!target!
!+ground danger+!	SYSVAL.ACPOSNREL	!ground pullup point!
!+GUNSSEL+!	DIAGIO	/GUNSSEL/
!+HG stage complete+!	STAGE	!HG stage! complete
!+high drag release+!	SYSVAL.WPNRLS	!high drag!
!+HL stage complete+!	STAGE	!HL stage! complete
!+HS stage complete+!	STAGE	!HS stage! complete
!+HUD slew legal+!	SYSVAL.SLEW	Section 4.3.1
!+HUDrefpt_az+!	SYSVAL.ACPOSNREL	Section 4.3.1
!+HUDrefpt_elev+!	SYSVAL.ACPOSNREL	Section 4.3.1
!+HUDREL+!	DIAGIO	/HUDREL/
!+IMS reasonable+!	SYSVAL.DEVREAS	!IMS Reasonable!
!+IMSAUTOC+!	DIAGIO	/IMSAUTOC/
!+IMSMODE eq Gndal+!	DIAGIO	/IMSMODE/
!+IMSMODE eq Grid+!	DIAGIO	/IMSMODE/
!+IMSMODE+! eq Iner+!	DIAGIO	/IMSMODE/
!+IMSMODE eq Magsl+!	DIAGIO	/IMSMODE/
!+IMSMODE eq Norm+!	DIAGIO	/IMSMODE/
!+IMSREDY+!	DIAGIO	/IMSREDY/
!+IMSREL+!	DIAGIO	/IMSREL/
!+in_flight+!	SYSVAL.VALSEL	/ACAIRB/
!+in pnl config+!	PNL.CONFIG	Section 4.6.1
!+input attempted+!	PNL.INPUT	Section 4.6.2
!+input requested+!	PNL.INPUT	Section 4.6.2
!+land based+!	PNL.INPUT	!Land!
!+land based pnl+!	PNL.INPUT	!Land!; !Data 23!
!+land velocity test failed+!	SYSVAL.IMSALN	!Land velocity test
!+land velocity test passed+!	SYSVAL.IMSALN	passed!
!+L-probe pnl+!	PNL.INPUT	!L-probe!; !Data 26!; see also Sections 4.6.2, 4.6.33
!+latitude+!	SYSVAL.ACPOSNREL	!latitude!
!+latitude_cup+!	SYSVAL.FIXLOCN	!Called-up point!
!+latitude error+!	SYSVAL.ACPOSNREL	Section 4.6.17
!+latitude pnl+!	PNL.INPUT	Sections 4.6.2, 4.6.5

Entry name	Producing Submodule	Mapping to Requirements
!+limited input+!	SUBRTN	see Sections 4.3.1,2,5
!+longitude+!	SYSVAL.ACPOSNREL	!longitude!
!+longitude_cup+!	SYSVAL.FIXLOCN	!Called-up point!
!+longitude error+!	SYSVAL.ACPOSNREL	Section 4.6.17
!+longitude pnl+!	PNL.INPUT	Sections 4.6.2, 4.6.5
!+low drag release+!	SYSVAL.WPNRLS	!low drag!
!+low lat ct a pnl+!	PNL.INPUT	Sections 4.6.2, 4.6.28
!+low lat ct b pnl+!	PNL.INPUT	Sections 4.6.2, 4.6.28
!+MA+!	DIAGIO	/MA/
!+mach reasonable+!	SYSVAL.DEVREAS	!mach unreasonable!
!+mag variation pnl+!	PNL.INPUT	!magnetic variation!; see also Sections 4.6.2, 4.6.9
!+map orient a pnl+!	PNL.INPUT	Sections 4.6.2, 4.6.29
!+map orient b pnl+!	PNL.INPUT	Sections 4.6.2, 4.6.29
!+mark+!	SYSVAL.FIXLOCN	!Mark number!; see also Section 4.6.15
!+mark lat+!	SYSVAL.FIXLOCN	Section 4.6.15
!+mark long+!	SYSVAL.FIXLOCN	Section 4.6.15
!+MFSW eq BOC+!	DIAGIO	/MFSW/
!+MFSW eq BOCOFF+!	DIAGIO	/MFSW/
!+MFSW eq CCIP+!	DIAGIO	/MFSW/
!+MFSW eq NATT+!	DIAGIO	/MFSW/
!+MFSW eq NATTOFF+!	DIAGIO	/MFSW/
!+MFSW eq TF+!	DIAGIO	/MFSW/
!+MULTRACK+!	DIAGIO	/MULTRACK/
!+nav_mode+!	MODE	Section 3.3
!+nav velocity test failed+!	SYSVAL.IMSALN	!Nav velocity test failed!
!+n coarse bias pnl+!	PNL.INPUT	Sections 4.6.2, 4.6.25
!+n coarse scale pnl+!	PNL.INPUT	Sections 4.6.2, 4.6.23
!+n fine bias pnl+!	PNL.INPUT	Sections 4.6.2, 4.6.26
!+n fine scale pnl+!	PNL.INPUT	Sections 4.6.2, 4.6.24
!+ND stage complete+!	STAGE	!ND stage!
!+ND2 stage complete+!	STAGE	end of !ND stage!
!+new align stage+!	STAGE	!New stage!
!+new (panel data item) entered+!	PNL.INPUT	Section 4.6
!+new test stage+!	STAGE	Section 3.7.0
!+oap ahead+!	SYSVAL.ACPOSNREL	!OAP!, !target!
!+offset brg pnl+!	PNL.INPUT	!offset bearing!; see also Section 4.6.13
!+offset dht pnl+!	PNL.INPUT	!Delta height!; see also Section 4.6.14
!+offset rng pnl+!	PNL.INPUT	!offset range!; see also Section 4.6.13

Entry name	Producing Submodule	Mapping to Requirements
!+OTS+!	SYSVAL.WPNRLS	!OTS!
!+panel error+!	PNL.INPUT	Section 4.6.2
!+PD stage complete+!	STAGE	!PD Tstage! complete
!+pnl input complete+!	PNL.INPUT	Section 4.6.2
!+PMDCTR+!	DIAGIO	/PMDCTR/
!+PMHOLD+!	DIAGIO	/PMHOLD/
!+PMLAND+!	DIAGIO	/PMLAND/
!+PMNORUP+!	DIAGIO	/PMNORUP/
!+PMSCAL eq 80+!	DIAGIO	/PMSCAL/
!+pnl config changed+!	PNL.CONFIG	Section 4.6.1
!+pnl config+!	PNL.CONFIG	Section 4.6.1
!+R65+!	SYSVAL.WPNRLS	!R65!
!+radalt priority pnl+!	PNL.INPUT	!Radalt!; !Data 24!; see also Sections 4.6.2, 4.6.31
!+radalt reasonable+!	SYSVAL.DEVREAS	!RADALT reasonable!
!+RE+!	DIAGIO	/RE/
!+rls_pts_passed+!	SYSVAL.WPNRLS	!rls pts passed!
!+rmax+!	SYSVAL.WPNRLS	!Rmax!
!+rmax+6000+!	SYSVAL.WPNRLS	!Rmax+6000!
!+rmin+!	SYSVAL.WPNRLS	!Rmin!
!+rmin+6000+!	SYSVAL.WPNRLS	!Rmin+6000!
!+SC stage complete+!	STAGE	!SC Tstage! complete
!+SINS dhdg pnl+!	PNL.INPUT	Sections 4.6.2, 4.6.8
!+SINS x offset pnl+!	PNL.INPUT	Sections 4.6.2, 4.6.7
!+SINS y offset pnl+!	PNL.INPUT	Sections 4.6.2, 4.6.7
!+SINS z offset pnl+!	PNL.INPUT	Sections 4.6.2, 4.6.8
!+SINS velocity test passed+!	SYSVAL.IMSALN	!SINS velocity test passed!
!+slew FLR delta az+!	SYSVAL.SLEW	!radar rate!
!+slew FLR delta rng+!	SYSVAL.SLEW	!radar rate!
!+slew HUD delta az+!	SYSVAL.SLEW	!HUD rate!
!+slew HUD delta elev+!	SYSVAL.SLEW	!HUD rate!
!+slew map delta lat+!	SYSVAL.SLEW	!Map rate!
!+slew map delta long+!	SYSVAL.SLEW	!Map rate!
!+special in range+!	SYSVAL.WPNRLS	!Special in range!
!+special solution+!	SYSVAL.WPNRLS	!Special Solution!
!+sr_ac_ap+!	SYSVAL.ACPOSNREL	!adjusted point!
!+sr_ac_bpup+!	SYSVAL.WPNRLS	!blast pullup point!
!+sr_ac_cup+!	SYSVAL.ACPOSNREL	!Called-up point!
!+sr_ac_ftpt+!	SYSVAL.ACPOSNREL	!Fly-to point!
!+sr_ac_gpup+!	SYSVAL.ACPOSNREL	!ground pullup point!
!+sr_ac_ip+!	SYSVAL.WPNRLS	!impact point!
!+sr_ac_oap+!	SYSVAL.ACPOSNREL	!OAP!, !target!
!+sr_ac_rls+!	SYSVAL.WPNRLS	!distance-to-release!
!+sr_ac_tgt+!	SYSVAL.ACPOSNREL	!target!

Entry name	Producing Submodule	Mapping to Requirements
!+sr reasonable+!	SYSVAL.DEVREAS	!Slant range reasonable
!+STA1RDY+!	DIAGIO	/STA1RDY/
!+STA2RDY+!	DIAGIO	/STA2RDY/
!+STA3RDY+!	DIAGIO	/STA3RDY/
!+STA6RDY+!	DIAGIO	/STA6RDY/
!+STA7RDY+!	DIAGIO	/STA7RDY/
!+STA8RDY+!	DIAGIO	/STA8RDY/
!+steering error to rls+!	SYSVAL.ACPOSNREL	Section 4.3.2
!+steering error to tgt+!	SYSVAL.ACPOSNREL	!target!
!+steering to tgt+!	SYSVAL.WPNRLS	!steering to target!
!+stik created+!	SYSVAL.WPNRLS	!rls pts passed!
!+stik empty+!	SYSVAL.WPNRLS	!rls pts passed!
!+stik_quan+!	SYSVAL.WPNRLS	!stik quantity!
!+symbol_az_on_ASL+!	SUBRTN	Section 4.3.11.2
!+target_in_range+!	SYSVAL.WPNRLS	!target-in-range!
!+TD+!	DIAGIO	/TD/
!+test stage+!	STAGE	Section 3.7.0
!+test_mode+!	MODE	Section 3.7
!+time_to prepare+!	SYSVAL.WPNRLS	Section 4.4
!+tgt ahead+!	SYSVAL.ACPOSNREL	!target!
!+time to ftpt+!	SYSVAL.ACPOSNREL	!Fly-to point!
!+TM stage complete+!	STAGE	!TM Tstage!
!+TOS+!	SYSVAL.WPNRLS	!TOS!
!+TS stage complete+!	STAGE	Section 3.2.0
!+update_mode+!	MODE	Section 3.5
!+v coarse bias pnl+!	PNL.INPUT	Sections 4.6.2, 4.6.25
!+v coarse scale pnl+!	PNL.INPUT	Sections 4.6.2, 4.6.23
!+velocity east system+!	SYSVAL.VALSEL	!System velocities!
!+velocity north system+!	SYSVAL.VALSEL	!System velocities!
!+velocity vertical system+!	SYSVAL.VALSEL	!System velocities!
!+weap_mode+!	MODE	Section 3.6
!+WEAPTYP+!	DIAGIO	/WEAPTYP/
!+wind dir+!	SYSVAL.VALSEL	Section 4.6.6
!+wind dir pnl+!	PNL.INPUT	Sections 4.6.2, 4.6.6
!+wind vel+!	SYSVAL.VALSEL	Section 4.6.6
!+wind vel pnl+!	PNL.INPUT	Sections 4.6.2, 4.6.6
!+wpns rlsd+!	SYSVAL.WPNRLS	!weapons released!
!+wmode_class+!	MODE	Section 3.6.16
!+x corr increm pnl+!	PNL.INPUT	Sections 4.6.2, 4.6.22
!+x drift pnl+!	PNL.INPUT	Sections 4.6.2, 4.6.19
!+y corr increm pnl+!	PNL.INPUT	Sections 4.6.2, 4.6.22
!+y drift pnl+!	PNL.INPUT	Sections 4.6.2, 4.6.20
!+z corr increm pnl+!	PNL.INPUT	Sections 4.6.2, 4.6.22
!+z drift pnl+!	PNL.INPUT	Sections 4.6.2, 4.6.21

ACKNOWLEDGMENTS

The following people have carefully reviewed this document for correctness:

Sandra Fryer (Naval Weapons Center, China Lake, CA);
Glenn Cooper (Vought Corporation);
Mark Jacobson (USAF)

Stuart Faulk, whose thoughtful analysis of the relationships among requirements, design, and implementation led to the current design of the Mode Determination submodule, should be considered a co-author of that chapter.

REFERENCES

[DI] Parker, Heninger, Parnas, Shore, Abstract Interface Specifications for the A-7E Device Interface Module; NRL Memorandum Report 4385; November, 1980.

[FD] Clements, Function Specifications for the A-7E Function Driver Module; NRL Memorandum Report 4658; November, 1981.

[MG] Britton, Parnas, A-7E Software Module Guide; NRL Memorandum Report 4702; December, 1981.

[REQ] Heninger, Kallander, Parnas, Shore, Software Requirements for the A-7E Aircraft; NRL Memorandum Report 3876; November, 1978.

[SO] Software Cost Reduction project, "Standard Organization for Specifying Abstract Interfaces", NRL Technical Memorandum, in progress. Until publication, readers are referred to the "Standard Organization" chapter of [DI].

FILME

0-8